Final Synopsis of the PhD Thesis to be submitted for the Award of Degree of Doctor of Philosophy in Computer Science

Machine Translation of Under-Resourced Languages: A Case Study of Translation from Kannada to Sanskrit and Tulu

By
Asha Hegde
Research Scholar
(Reg. No. MU/EXB/PhD/CR.38/C.S./2021-22/E.25)

Under the Guidance of
Dr. H. L. Shashirekha
Professor

Post Graduate Department of Studies and Research in Computer Science
Mangalore University, Mangalagangothri, Konaje
Karnataka, India
May 2025

Contents

1 Introduction **4**
1.1 Motivation 4

2 Related Work **6**

3 Objectives **8**

4 Proposed Work **8**
4.1 Development of Kannada-Sanskrit Machine Translation Systems 8
4.1.1 *KanSan* Parallel Corpus Construction 9
4.1.2 Kannada-Sanskrit Machine Translation Baselines 11
4.1.2.1 Pre-processing 11
4.1.2.2 Subword Tokenization 12
4.1.2.3 Statistical Machine Translation 12
4.1.2.4 Neural Machine Translation 12
4.1.3 Text Augmentation for *KanSan* Corpus 15
4.1.4 Experiments and Results 16
4.2 Development of Kannada-Tulu Machine Translation Systems 17
4.2.1 Construction of Kannada-Tulu Parallel Corpus - *KT1* 18
4.2.2 Automatic Text Generation for Kannada-Tulu Parallel Corpus - *KT1* 19
4.2.3 Limitations of *KT1* Corpus 20
4.2.4 Construction of Kannada-Tulu Parallel Corpus - *KT2* 20
4.2.5 Automatic Text Generation for Kannada-Tulu Parallel Corpus - *KT2* 21
4.2.5.1 Evaluation of the Text Generated by ATG for *KT2* 23
4.2.6 Kannada-Tulu Machine Translation Baselines 25
4.2.7 Experiments and Results 25
4.3 Syllable-level Morpheme Segmentation and Boundary Feature Integration for Kannada–Tulu Neural Machine Translation 26
4.3.1 Proposed Method for Morpheme Boundary Detection 29
4.3.1.1 Feature Extraction 29
4.3.1.2 Model Construction 30
4.3.2 Experiments and Results for Morpheme Boundary Detection 30
4.3.3 Proposed Method for Integrating Morpheme Boundary Features to Neural Machine Translation 31
4.3.4 Experiments and Results of Morpheme Boundary Integration 32
4.4 Development of Part-Of-Speech Tagger and Integration of Part-Of-Speech Features for Neural Machine Translation 32

4.4.1 Pre-processing . . . 32
4.4.2 Annotation . . . 32
4.4.3 Corpus Statistics . . . 33
4.4.4 PosT: Proposed Method for Building Tulu Parts-Of-Speech Tagger . . . 33
4.4.4.1 PosT_ML . . . 34
4.4.4.2 PosT_CRF . . . 35
4.4.4.3 PosT_DL . . . 36
4.4.4.4 Post_TL . . . 38
4.4.5 Experiments and Results for Parts-Of-Speech Tagger . . . 43
4.4.6 Proposed Method for Integrating Parts-Of-Speech Features to Neural Machine Translation . . . 43
4.4.7 Experiments and Results of Parts-Of-Speech Integration . . . 44
4.5 Exploration of Orthographies for Enhancing Neural Machine Translation of Under-resourced Languages . . . 44
4.5.1 Bilingual Parallel Corpus Construction . . . 45
4.5.2 Test Sets to Assess Bilingual MT Models . . . 47
4.5.3 Construction of Multilingual Parallel Corpus . . . 47
4.5.4 Methodology for Translation . . . 47
4.5.5 Experiments and Results . . . 49
4.6 Bridging Under-resourced Languages with Semantic Multilingual Neural Translation . . . 55
4.6.1 Construction of Multilingual Parallel Corpus . . . 56
4.6.2 Methodology for Translation . . . 58
4.6.3 Experiments and Results . . . 60
4.7 Automatic Post Editing for Neural Machine Translation . . . 60
4.7.1 Proposed Method . . . 60
4.7.2 Experiments and Results . . . 61
4.8 Performance Comparison . . . 62
4.8.1 Kannada-Sanskrit Machine Translation Systems . . . 63
4.8.2 Kannada-Tulu Machine Translation Systems . . . 63

5 Conclusion and Future Works **64**

6 Publications **65**
6.1 Publications in Book Chapter . . . 65
6.2 Publications in International Conferences . . . 65
6.3 Papers Under Communication . . . 65

7 Papers Published in the same Domain **66**

8 Organization of the Thesis **66**

1 Introduction

Machine Translation (MT) is the automated translation of source language text into target language text preserving the context of the source language text [1]. It is the most complicated area of Natural Language Processing (NLP) which is a sub field of Artificial Intelligence (AI) that enables the machines to interpret, understand and extract meaning from natural languages [2]. Significance of MT in everyday lives is in various fields such as preference for native language, enrichment of the global economy, opportunities to spread knowledge, etc. The key to the popularity of MT systems is its speed compared to that of human beings [3]. MT systems are giving considerable accuracy and are employed in diverse business areas, such as in business to localise the domain knowledge, in Software as a Service (SaaS) based MT platforms, in the Government sector for translating confidential information especially in defence, in wearable voice translators, in speech-to-speech translator application for different language pairs, in medical translator applications, in government gadgets and notifications, to perform real-time translation, and so on.

Out of more than 7000 languages in the world, only 20 languages, such as English, Spanish, German, are predominantly spoken by several communities and the availability of digital dictionaries, lexicons, parallel corpora, and computational tools for these languages have made them resource-high [4]. Except these 20 languages, other languages are spoken by fewer people when compared to the overall population. Further, lack of proper corpora, tools, machine readable dictionaries, etc., have made these languages resource-poor. The number of resource-high languages is very small when compared with the languages in use. This indicates the importance of MT systems that aims to facilitate knowledge and to bridge the gap between resource-high language and resource-poor languages.

1.1 Motivation

India is a multilingual country with 22 scheduled languages; specifically categorised between Indo-Aryan languages and Dravidian languages. Though these two language groups have some mutual influences and the core vocabulary of these two language families is different. Dravidian languages are spoken in the south part of India by 215 million people [5]. It has around 26 languages, including Tamil, Malayalam, Kannada, and Telugu, which are the major languages in the Dravidian language family, and Tulu and Kodava are small literary languages. Dravidian languages have a huge amount of loan words which are borrowed from Sanskrit and Sanskrit grammar written by Panini is considered as base for the grammar of Dravidian languages. Hence, Dravidian languages share many of the linguistic features and vocabulary with Sanskrit. These languages have their own writing script that has been allotted a separate block in the Unicode computing industry standard. Major similarity between these languages is that all these languages are written from left to right and maintain

alike features in the word formation and sentence construction. These languages consist of sequence of characters that forms words, words are constructed by adding prefixes and/or suffixes to the root word and sequence of words will form a sentence. The Dravidian languages follow alpha-syllabic writing scheme and individual symbols are called as syllables [6]. These languages share similar linguistic features and few of them are listed below [5]:

- Verbs have a negative as well as an affirmative voice
- Root word can be extended using one or more suffixes or prefixes according to the comfort level of the speaker
- Phonology of all Dravidian languages follow similar strategy
- Though these languages follow Subject-Object-Verb word order, they are considered as free word order languages as meaning of the sentence will not change when the word order is changed
- Gender classification is made based on suffix
- Nouns are declined, showing case and number
- These languages have their own alphabets related to the Devanagari alphabet that is used for Sanskrit

Sanskrit language popularly known as Devabhasha, has a history of more than 3500 years [6] and it is written in Devanagari script. Sanskrit belongs to the Indo-European language and the visible impact of Sanskrit is found in all most all Indian languages. As many languages have originated from Sanskrit, learning Sanskrit provides a greater understanding of other languages. Therefore, Sanskrit is considered as the mother of all languages. Though Sanskrit has significant history, Sanskrit has been spoken by only around 2.5 million people as their first, second and third language.

Kannada is the second oldest Dravidian language spoken predominantly by the people of Karnataka and it is written in Brahmi script. There are about 44 million Kannada speakers throughout the world and it is also spoken as a second and third language by over 12.6 million non Kannada speakers in Karnataka. Tulu is one of the major Dravidian languages predominantly spoken by the people of Dakshina Kannada, Udupi and some places of Kasaragod and it is written in Tigalari script. More than 2.5 million people speak Tulu and they consider it as their mother tongue. Tulu speaking community with its distinct sociocultural traits, religious practices, artistic traditions and theatrical forms has made significant contribution to the cultural heritage of Karnataka and through it to the totality of Indian culture and civilization.

Though Kannada, Sanskrit and Tulu have rich collections of articles, they are still considered as under-resourced languages due to the availability of very less digital resources and

tools for processing these languages. In this regard, Language Translation (LT) technology gains importance as it provides the easy means of communication between people when language barrier is a major issue. Various LT technologies are available, namely, manual translation or human translation, Machine aided translation and automatic or MT. As MT is fast, inexpensive, reliable and consistent compared to other LT technologies, it is gaining popularity.

The translation of information available in a regional more established language like Kannada to an ancient language Sanskrit and an ancient regional language like Tulu helps to gain the attention of researchers in the language technology tasks using these languages. This also assures the necessity of exploring efficient techniques for handling these languages for translation. Further, the parallel corpus for these language pairs are digitally zero-existed for translation and hence the preparation of annotated corpus itself will be a contribution. Translation of texts belonging to the above under-resourced languages promote resource preservation.

2 Related Work

In the recent decade, MT techniques in general and NMT approaches in particular have improved quickly with many practical applications. This visible growth in MT research is being carried out particularly for high-resource languages, such as English, German, French, Spanish, and Chinese [7]. An NMT model requires a parallel corpus of good quality with sufficiently large size in order to obtain state-of-the-art results [8]. However, most of the languages in the world are under-resourced languages that usually face lack of parallel corpus issues and few languages do not exist in digital form itself [9].

Translation between morphologically rich languages, such as Indian languages in general and Dravidian languages in particular, is still considered as uncommon. Further, the under- resource issues of these languages intensify the translation complexities. From the literature, it is clear that very few attempts were made to develop MT systems between morphologically rich agglutinative under-resourced languages. In addition, translation among Kannada-Sanskrit (Kan-San) and Kannada-Tulu (Kan-Tul) language pairs are rarely explored. Hence, this fact will add more scope in the translation of Kan-San and Kan-Tul language pairs.

Parallel corpora are crucial for developing effective MT systems, especially for under-resourced languages like Kannada, Sanskrit, and Tulu. However, these languages face a significant challenge due to the lack of sufficient parallel corpora, hindering the progress of MT systems for these language pairs. The conventional method of constructing a parallel corpus relies on manual translation with the help of linguists [10]. While this approach yields high-quality parallel data, it is expensive and laborious. Hence, MT researchers have turned their focus towards human-assisted or automatic methods for parallel corpus construction,

aiming to streamline the process and reduce the associated expenses [11]. In this direction, web crawling [12] and web scraping techniques [13], are found to be more promising for constructing parallel corpora for under-resourced languages by automatically extracting bilingual content from the online resources. In addition, leveraging technologies such as the Google Translation Application Programming Interface (API) [14], parallel text generation from speech by performing speech-to-text translation [15], aligned subtitles from the movies or videos [16] and crowd-sourced translations [17], aid in the construction of parallel corpora. These automated and semi-automated approaches can significantly enhance the corpus creation process. However, their effectiveness relies on post-processing for quality control to ensure accurate and reliable parallel text alignments [18].

In addition to manual and web-based methods for constructing parallel corpora, the Text Augmentation (TA) approach that comprises a dictionary-based word induction is also explored by the scholarly community [19] to enhance the size of the parallel corpus. In this method, words in an under-resourced language are replaced with similar words from a high-resource language within a sentence with the help of a bilingual dictionary. These modified sentences are subsequently refined using an unsupervised MT framework to produce augmented text data, presenting a practical solution for enriching parallel corpora to address resource issues in under-resourced languages. In addition, various TA techniques, including back-translation [20], dictionary-based synonym replacement [21], word embeddings-based synonym replacement [22], text paraphrasing [23], knowledge injection [24], and data generation through rule-based methods [25], are also explored to enhance the size of the parallel corpora.

In recent decades, a number of research articles have explored the impact of multilingual approaches in MT where models are trained on a corpus that contains parallel text in multiple languages, allowing them to learn the relationships between different languages simultaneously [26]. Studies have highlighted the benefits of leveraging multilingual data to enhance translation quality across various language pairs, exclusively for under-resource languages [27]. Multilingual MT systems have emerged as a focal point for MT researchers, particularly in under-resourced language settings [28] and have shown considerable promise in addressing the scarcity of data by leveraging multilingual corpora and shared representations across languages. Through their ability to learn from multiple language pairs simultaneously, multilingual MT systems have demonstrated improved translation quality compared to bilingual MT systems where MT systems are trained using the source text belonging to one language and target text belonging to another language. The literature also highlights that script normalization [29] and the integration of linguistic features [30] can significantly improve the performance of MT systems. Additionally, automatic post-processing of translations generated by the MT models has been shown to further enhance their quality [31].

The literature review reveals that MT for Kan-San and Kan-Tul language pairs is rarely explored due to the lack of bilingual parallel corpora for these languages and the works

which are reported use very small data [30]. It also suggests that multilingual MT models offer more benefits than bilingual systems, especially when dealing with under-resourced languages. Additionally, incorporating techniques such as script normalization, linguistic features, and automatic post-processing has shown a significant improvement in translation quality. These methods are particularly effective for under-resourced languages, enhancing both the accuracy and fluency of the translations.

3 Objectives

These observations have motivated us to propose the NMT from Kannada to Sanskrit and Tulu, the under-resourced languages with the following objectives drawn for the proposed work:

- Construction of a bilingual parallel corpora for Kan-San and Kan-Tul language pairs
- Developing additional pre-processing technique for Kannada to Sanskrit and Kannada to Tulu MT systems
- Developing MT systems for Kannada to Sanskrit and Kannada to Tulu
- Developing effective post-processing technique for Kannada to Sanskrit and Kannada to Tulu MT systems

4 Proposed Work

To achieve the need to design an efficient MT systems for Kan-San and Kan-Tul language pairs, we constructed properly aligned bilingual parallel corpora for these languages. Because, construction of bilingual parallel corpora is the fundamental step in the development of MT systems. Using these parallel corpora, we implement baselines to translate Kannada sentences to Sanskrit and Tulu sentences. Further, from the literature survey it is evident that pre-processing techniques namely, sandhi split, handling rare words, etc., in addition to conventional pre-processing techniques shows better translation performance [32]. With this view, this research work also proposes effective pre-processing and post-processing technique to enhance the translation performance of Kan-San and Kan-Tul MT systems.

4.1 Development of Kannada-Sanskrit Machine Translation Systems

Datasets play a crucial role in evaluating algorithms for research, but most benchmarked datasets are available for high-resource languages like English, German, and French, leaving

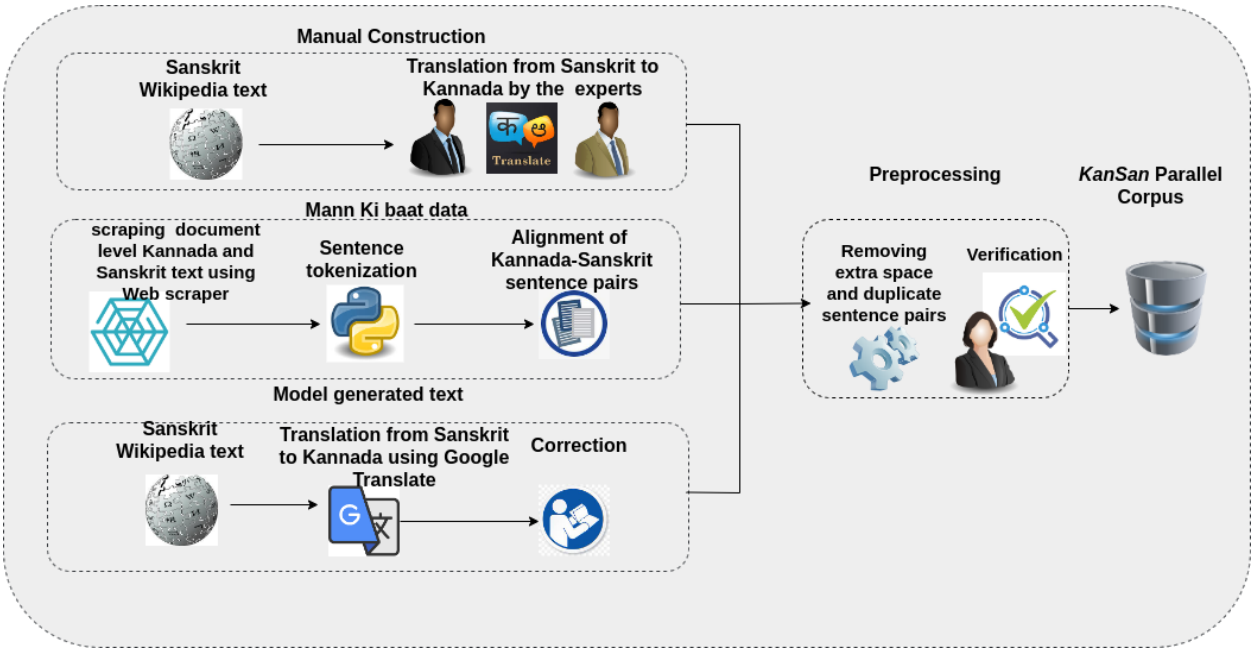


Figure 1: Framework of the *KanSan* parallel corpus construction

under-resourced languages such as Kannada, Tamil, Hindi, Sanskrit, Malayalam, and Telugu under-represented. This has led to the creation of datasets for NLP applications like MT, focusing on under-resourced morphologically rich languages. In this context, the Kan-San parallel corpus are developed for building the Kan-San MT system. Additionally, a morpheme boundary detection and segmentation dataset is created for Kannada and Tulu morphological segmentation and generation. Furthermore, a Tulu POS-tagged corpus is constructed for the development of a Tulu POS tagger. These efforts aim to advance NLP technologies for these languages, addressing the challenges posed by their under-representation in digital resources.

4.1.1 *KanSan* Parallel Corpus Construction

In view of the lack of parallel corpora for Kan-San MT, *KanSan* - a Kannada-Sanskrit parallel corpus is constructed. In most of the cases, parallel corpora involve monolingual text in each language of the language pair [10]. Framework for the construction of *KanSan* parallel corpus is shown in Figure 1.

The first step in the construction of *KanSan* corpus is extracting monolingual Sanskrit text from Wikipedia[1] by web-crawling[2]. The text collected from Wikipedia dumps have lot of noise, such as empty lines, lines consisting of only digits, symbols, English letters, and urls. This noise is removed from the text using handcrafted code[3] subjected to sentence

[1] https://sa.wikipedia.org/wiki/
[2] https://github.com/hegdekasha/wikiextractor
[3] https://github.com/hegdekasha/wikiextractor/wikipreprocessing

tokenization using indicNLP library[4] which resulted in 9,38,184 Sanskrit sentences. Part of this data is used in the manual construction and Model Generated Text (MGT) approaches to create *KanSan* corpus. The approaches used to create *KanSan* corpus are described below:

i) Manual Construction - 25,000 Sanskrit sentences are randomly selected from cleaned Sanskrit Wikipedia text which are then manually translated into Kannada with the help of three native Kannada speakers who have expertise in Sanskrit. Each sentence is translated by a minimum of two translators according to the following guidelines provided to them:

- The context of the source sentence must be clearly conveyed in the target sentence
- During translation, the adequacy and fluency of the target language must be considered
- Words in the source sentence that do not belong to the source language must be entered in the target sentence in the same form as they appeared in the source sentence
- Digits must be written either using the target language phonetics or in the number format

Using these guidelines, each sentence is translated in two phases: i) all the sentences are translated by one or the other translator ii) the translator who is not involved in the first phase verifies and corrects the sentences in the second phase.

ii) Model Generated Text - uses pre-trained models or open sourced services/API to generate text [14]. Motivated by this, MGT is considered as one of the sources of data in the construction of *KanSan* corpus using Google Translate[5] (GT) - an open sourced translation API. Recently GT added Sanskrit to its system which provides decent translations from Sanskrit text to Kannada text. 25,000 non-overlapping Sanskrit sentences randomly selected from the cleaned Sanskrit Wikipedia text obtained earlier are translated to Kannada text by the GT API. The output of GT is manually verified by the experts to check the fluency and adequacy of the target language and corrections are made wherever required.

iii) Mann Ki Baath Data - *Mann Ki Baath* online repository[6] contains multilingual parallel data of 13 Indian languages including Kannada and English. A web-scraper[7] is used to extract Kannada text from this repository at the document level. Further, Sanskrit text of *Mann Ki Baath* data which is also at the document level is obtained from the online Sanskrit documents website[8]. Both Kannada and Sanskrit documents are temporally aligned and are then tokenized using the indicNLP library followed by the manual verification of sentence alignment. This process yields 23,758 Kannada-Sanskrit sentence pairs. It may be

[4] https://github.com/anoopkunchukuttan/indic_nlp_library
[5] https://translate.google.co.in/
[6] https://www.pmindia.gov.in/en/mann-ki-baat/
[7] https://github.com/MSKantulu/Mann-Ki-Baat
[8] https://sanskritdocuments

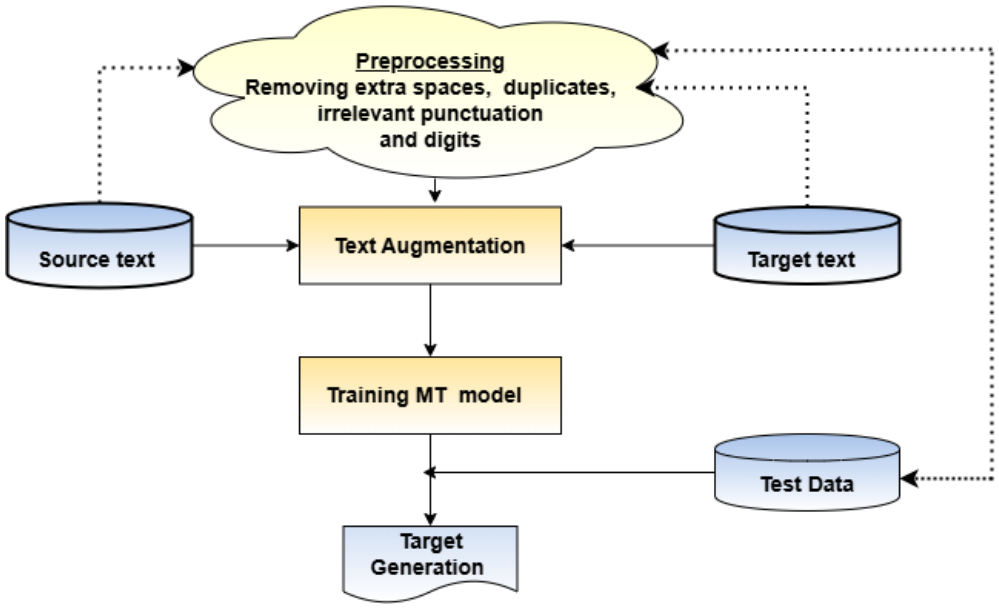


Figure 2: Frame work of the proposed methodology for implementing baselines

noted that the verification of sentence alignment has consumed a significant amount of time as the parallel text is temporally aligned. The sentence alignment has the significant role in building a comprehensive parallel corpus and hence on the quality of MT. These three approaches together yielded 73,758 Kannada-Sanskrit sentence pairs.

4.1.2 Kannada-Sanskrit Machine Translation Baselines

The primary objective of the proposed methodology is to establish baselines by implementing a wide range of MT models and benchmarking the newly constructed parallel corpora. The methodology involves both SMT and various encoder-decoder-based NMT approaches. As no research work has been reported specifically for Kan-San and Kan-Tul language pairs, the existing NMT models which are explored for similar research works are considered as baselines. Therefore, inspired by Premjith et al., [33] and Choudhary et al., [34], the proposed baselines are implemented. These NMT models include RNN with GRU and LSTM units, BiRNN with GRU and LSTM units, as well as transformers with GRU and LSTM units. Additionally, sub-word tokenization technique is explored, as it has shown promising results in similar research studies. The general framework of the proposed methodology is illustrated in Figure 2.

4.1.2.1 Pre-processing

The impact of pre-processing in improving the quality of MT output has been reported in several research works [35]. Apart from removing the noise that reduce the quality of the corpus, pre-processing will also evaluate the corpus semantically, stimulating a tendency for

the creation of sentence alignments. Hence, during pre-processing, extra space such as initial space, end space, empty lines, double white space, and duplicate sentence pairs are removed from the parallel corpora.

4.1.2.2 Subword Tokenization

Subword tokenization is a popular tokenization method in which rare words are split into the most frequently occurring words and are represented as a pair of bytes in a sequence with an unused byte [36]. Usage of subword tokenization is intended to avoid Out-Of-Vocabulary (OOV) issues that occur when a word is present in the Test set but not in the Train set. Further, OOV issue occurs specifically in morphologically rich languages due to lengthy morphemes. Byte Pair Encoding (BPE)[9] - a popular subword tokenization technique developed based on data compression algorithm is used in this work to obtain subword units. In BPE, the frequent words are represented by a single token along with a special symbol (_) and the rare words are broken down into two or more subword tokens. OOV words will be broken down into more frequently occurring subwords. It may be noted that, in this work, BPE with a vocabulary size of 50,000 for both Kannada and Sanskrit is used in building vocabulary for both source and target text which are then transformed into vectors to train the transformer-based models.

4.1.2.3 Statistical Machine Translation

Based on Bayes theorem, Brown et al. [37] developed a statistical model for MT using parallel corpus. In this approach, translation is done based on the probability distribution of sentences in the parallel corpus. SMT system has two phases. In the first phase, distribution of the likelihood over sequence of terms is computed using the Language Model (LM), which gives the distribution of target sentence viz. $p(t)$ where 't' represents the target sentence. In the second phase, the Translation Model (TM) gives the probability of source sentence given by the target sentence viz. $p(s \mid t)$ where 's' represents the source sentence. Searching for the best translation is done based on the maximum probability value.

4.1.2.4 Neural Machine Translation

NMT is one of the corpus based approaches to carry out translation using Neural Networks (NN) and it is considered as the most suitable approach to perform sequence-to-sequence translation at sentence level [38]. The model is based on encoder-decoder architecture with attention mechanism [39] and is further extended with transformer [40] - a self-attention architecture.

Sequence-to-sequence architecture is essentially used for sequence prediction and is also known as Vanilla RNN. This conditional probability distribution is modeled using en-

[9] https://bpemb.h-its.org/

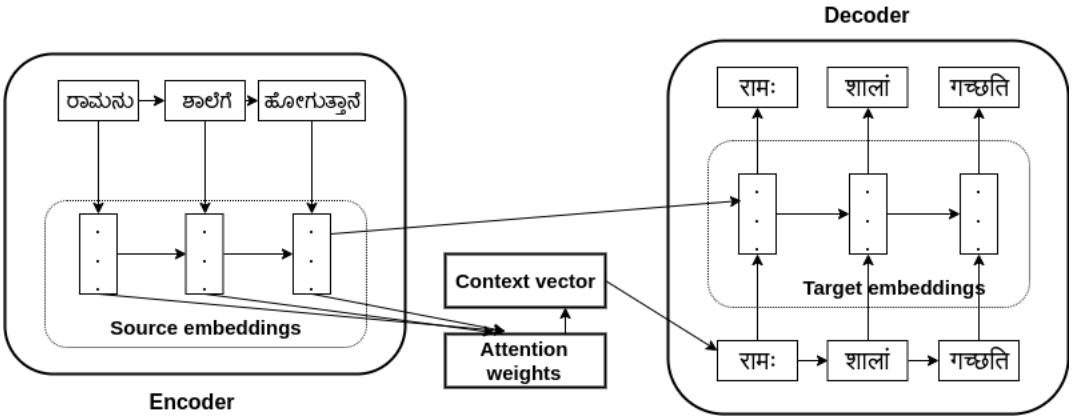


Figure 3: Encoder-decoder based Neural Machine Translation model

coder–decoder architectures and the basic encoder-decoder architecture is shown in Figure 3. This architecture consists of two parts, namely: (i) encoder which accepts source language text as input and produces its intermediate representation (ii) decoder which generates the output based on the encoding vector and previously generated words. Suppose, S is the source text and T is the target text, encoder transforms the source sentences s_1, s_2, ... , s_m into fixed dimension vectors S_1, S_2, ... , S_M and the decoder uses conditional probability to produce the output word-by-word. Using the chain rule, equation 2 is derived from the equation 1. While decoding, next word is predicted using previously predicted word vectors and source sentence vectors as in equation 2. Each term in the distribution is represented with a softmax over all the words in the vocabulary [41].

$$P(T \mid S) = P(T|S_1, S_2, ...S_M) \tag{1}$$

$$P(T \mid S) = P(t_i \mid t_0, t_1, t_2, ...t_{i-1}; s_1, s_2, ...s_m) \tag{2}$$

BiRNN are the NNs which allow both forward and backward traversal of input. Regular RNNs have the disadvantage of reading the input sequence in only one direction, usually from past to future. BiRNNs were proposed by Schuster et al. [42] to capture past and future contexts using two independent RNNs. The idea is to have two independent recurrent layers: one for processing the input sequence in a forward direction and the other for processing the input sequence in backward direction.

Transformer is a self-attention-based model with an innovative multi-head attention mechanism that has gained the popularity in recent years. For many tasks, such as MT, spell check and topic classification, transformer has become the state-of-the-art model. Transformer-based NMT models, in comparison to other NMT models not only provide better translation results but will also be trained in less time [43]. These models are composed of encoder and decoder modules. Equation 3 presents the mathematical formulation

of the optimization criterion for training the transformer model given a reference target sentence containing tokens [44].

$$L = \frac{1}{T} \sum_{m=1}^{T} \sum_{n=1}^{\Psi} (d_{mn} * log(P_{mn})) \tag{3}$$

In equation 3, P_{mn} - probability distribution of the output, T - tokens of target sentences and d_{mn} - whether n^{th} token in the dictionary is the true value at the m^{th} position in the target sentence.

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) specifically designed to capture long-range dependencies in sequential data. Unlike traditional RNNs, LSTMs use memory cells and gating mechanisms—namely input, output, and forget gates—to regulate the flow of information and avoid issues like vanishing or exploding gradients. This makes LSTMs highly effective for tasks involving time-series data, natural language processing, and machine translation. Their ability to retain and selectively update contextual information over long sequences has made them a foundational component in many deep learning architectures prior to the rise of Transformer models [45].

Gated Recurrent Unit (GRU) is a type of Recurrent Neural Network (RNN) designed to efficiently capture dependencies in sequential data, similar to Long Short-Term Memory (LSTM) networks but with a simpler architecture. GRUs combine the forget and input gates into a single update gate and use a reset gate to control the flow of information, allowing them to retain relevant context over time. Due to their reduced complexity and fewer parameters, GRUs are often faster to train and can perform comparably to LSTMs in many sequence modeling tasks. They are widely used in applications such as speech recognition, text generation, and neural machine translation [46].

Encoder - Basically, encoders transform sentences into vector values, which represent their context. It is necessary to obtain word representations, often known as word embeddings, of source language words before feeding them into the encoder network. Encoder network transforms these word embeddings into sentence embeddings thereby generating the context vector consisting of the context of the source sentence [38]. By convention, the encoder network can be constructed using RNN and can also be constructed either using BiRNN [33], or transformers [40]. Unlike RNN, BiRNN has two RNNs to compute forward and backward hidden state sequences [42]. Further, the transformer-based encoder network extracts the features of all words in the source sentence using self-attention mechanism.

Decoder - The decoder network is initialized by the final layer of the encoder network ie. context vector and the decoder network can be either RNN, BiRNN, or transformer based on the type of the encoder network. The target language words (tgt) are predicted based on the sentence embeddings obtained at the encoder, previously predicted target words and the contextual information of each source word. From equation 2, $p(b_k|b_{k_{-1}}, ..b_1, a, c) = tgt$

Table 1: Statistics of the *KanSan* parallel sentences collected from different data sources

Data source	# of Kannada-Sanskrit sentences extracted
Manual Translation	25,000
Mann Ki Baat	23,758
Model Generated Text	25,000
Total (before pre-processing)	73,758
Total (after pre-processing)	72,671
Back-translation	27,500
Total	1,00,171

Table 2: Statistics of the *KanSan* parallel corpus and details about Train and Test set

Languages	**# of Tokens**	**# of Unique words**	**Average words in a sentence**
Sanskrit	3,89,556	51,259	9
Kannada	3,57,117	50,953	11
Total # of Sentences	1,01,171		
Language pair	**Train set**	**Test set**	
san-kan	1,00,171	1,000	

can be obtained as,

$$tgt = g(b_{k_{-1}}, f(h_{k-1}, k_{-1}, c_k), c_k) \tag{4}$$

where, g is a nonlinear function, b_{k-1} is the previously predicted target word, c_k contextual information of each word, and h_{k-1} is the hidden state in the decoder at (k-1) step.

4.1.3 Text Augmentation for *KanSan* Corpus

The *KanSan* parallel corpus where major part of the data is constructed manually, which is the traditional way of corpus construction. In addition, MGT and effective utilization of Mann Ki Baath data along with manual verification are also explored. While, these approaches include significant involvement of linguists which is expensive and laborious. Hence, Back-translation (BT) - one of the simplest and more effective TA techniques is employed to increase the size of *KanSan* corpus and the description is given below:

Back-translation - exploits the monolingual data for translation and it can be used to increase the size of the parallel corpus by translating the target language text into the source language text creating synthetic data [36]. 72,671 Kan-San sentence pairs (T_c) out of 73,758 obtained as mentioned in Section 4.1.1 after removing duplicate sentences and short sentences (sentences with less than 4 words) are used for BT. Out of these 72,671 sentence pairs in T_c, 10,000 sentence pairs are chosen randomly and the Kannada sentences

Table 3: Hyper-parameters and their values used in NMT model

Hyper-parameters	**Values**
word vector size	512
encoding layers	2
decoding layers	2
heads	8
learning rate	1.0
dropout	0.3
batch size	64
train steps	1,00,000
encoder type	transformer
rnn type	lstm
position encoding	True
optimization	sgd
check-point	10,000
maximum-sequence-length-src	default
maximum-sequence-length-tgt	default

from these pairs are used to test and the remaining sentence pairs are used to train the Transformer+BPE model. The steps that are used to carry out BT are given below:

1. 10,000 Kannada (source) sentences in the Test set are translated into Sanskrit (target) sentences

2. The reverse translation is carried out by training the Transformer+BPE model using the interchanged Kannada (source) and Sanskrit (target) language text (obtained in step 1)

3. The Sanskrit (target) sentences that were obtained in step 2 are considered as the Test sentences for reverse translation to generate Kannada sentences (source)

4. To overcome the biasing, Steps 1, 2, and 3 are repeated four times with non-overlapping Test sets to get the Kannada (source) sentences in parallel to the Sanskrit (target) sentences

The Kan-San sentence pairs obtained by BT are manually verified and added to T_c. Further, duplicate sentence pairs in the new T_c are removed to get the final corpus. Thus, BT contributed 27,500 Kan-San parallel sentences in the construction of *KanSan* corpus resulting in 1,01,171 parallel sentences. The number of parallel sentences obtained from each data source including BT and the statistics of *KanSan* corpus are given in Tables 1 and 2 respectively.

4.1.4 Experiments and Results

This section presents the experiments and results for the *KanSan* parallel corpus, both before and after TA. The findings highlight the impact of TA on model performance across

Table 4: Performance measure of the San-Kan MT models in terms of BLEU score before and after ATG

Models	Kan-San		San-Kan	
	BLEU	CHRF	BLEU	CHRF
SMT (before ATG)	03.84	16.90	04.06	17.50
SMT (after ATG)	04.53	18.02	05.42	19.70
RNN (before ATG)	06.48	20.00	08.32	21.63
RNN (After ATG)	07.20	21.40	09.30	22.07
BiRNN	07.32	20.35	09.93	24.81
Transformer	08.53	24.09	11.56	28.06
Transformer+BPE	**09.84**	26.37	**12.63**	30.80
Google translator	07.13	20.91	12.28	29.00

these datasets. NMT models for translating Kan-San sentence pairs and vice versa are implemented using the open-source NMT framework, OpenNMT-py[10]. This framework offers a structured encoder-decoder architecture enriched with attention mechanisms, allowing seamless handling of sequence-to-sequence prediction tasks [47]. The hyper-parameters and their values used to build NMT models that gave the best results are shown in Table 3.

Selection of evaluation metrics is a crucial component for the evaluation of any algorithm and BLEU score is used for evaluating the proposed models. BLEU score is the popular evaluation metric which is evident by the literature. It is calculated by counting matching word n-grams between the candidate translation and the reference text [48]. The main idea of using BLEU score is to evaluate MT systems as close as possible to human evaluation. Further, Character F-score (CHRF) scores are also used to evaluate the translation performances of the proposed models, where Character-level n-gram precision and recall are used to compute the CHRF score.

Performance measure of MT models are given in Table 4 indicates that all the NMT models performed better than SMT model in terms of BLEU scores indicating that the NMT models are always winners over SMT models in sequence prediction task [49]. BiRNN model has shown slight improvement in the BLEU scores for both Kan-San and San-Kan translations over RNN baseline. Transformer-based models due to its self-attention mechanism, outperformed BiRNN with BLEU scores of 1.21 and 1.63 for Kan-San and San-Kan translations respectively.

4.2 Development of Kannada-Tulu Machine Translation Systems

Over the past decade, NMT has experienced remarkable growth and is now reaching a phase of maturity. Despite its popularity and widespread use, NMT still performs poorly on under-resourced language pairs due to the lack of sufficient parallel corpora, especially when compared to high-resource languages. In light of the ongoing advancements in machine

[10] https://github.com/OpenNMT/OpenNMT-py

translation, this work explores practical approaches for developing MT systems between Kannada and Tulu. Given the absence of existing parallel corpora for this language pair, the primary objective of this work is to create a parallel corpus to support effective translation and further research in this domain.

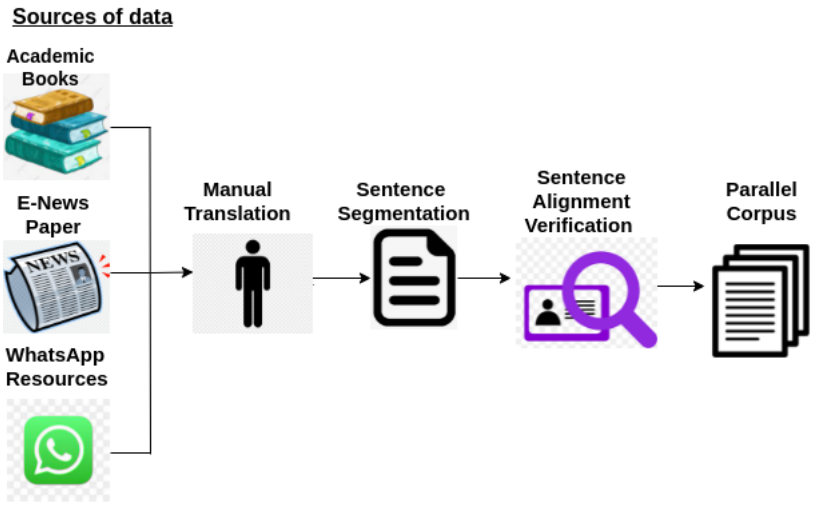


Figure 4: Workflow of parallel corpus construction for Kannada-Tulu language pair

4.2.1 Construction of Kannada-Tulu Parallel Corpus - *KT1*

As per the available literature, Kannada-Tulu parallel corpus for MT is not available. Hence, the construction of Kan-Tul parallel corpus (*KT1*) has been started from scratch. Figure 4 illustrates the workflow of building a Kan-Tul parallel corpus and the statistics of Kannada-Tulu parallel corpus is summarized in Table 5.

Sources of data which are used to extract the Tulu text are explained below:

- **Academic books** - The of 7^{th} and 8^{th} standard Tulu text books of Government of Karnataka are used to collect Tulu text[11]. As these texts are in pdf format, they are converted into an editable form using NewOcr - an open source Optical Character Recognition (OCR) tool[12]. The editable text subjected to sentence segmentation yields 2,500 Tulu sentences.

- **Daily newspaper** - 2,500 Tulu sentences are collected from daily Tulu e-newspaper[13] which focuses more on local news.

- **Tulu magazine** - 2,500 Tulu sentences are collected from a Tulu monthly magazine[14] which consists of Tulu articles from Tulu writers and public reviews on current issues.

[11]https://www.governmentexams.co.in/
[12]toolhttps://www.newocr.com/
[13]https://www.timesofkudla.com/
[14]https://poovari.business.site/

Table 5: Statistics of Kan-Tul parallel corpus

Languages	# of Tokens	# of Unique words	Average # of words in a sentence
Kannada	35,622	6,729	7
Tulu	36,139	7,912	8
Total # of Sentences	10,000		

- **WhatsApp resources** - In recent years social media is playing a major role in developing resources for under-resourced language like Tulu in terms of comments, reviews, and articles. To construct Tulu monolingual corpus, 1,760 Tulu articles on Tulu culture, history, and short stories forwarded in WhatsApp groups are collected.
- **Other sources** - 740 Kannada-Tulu parallel sentences are extracted from examples given in online Tulu dictionary[15].

Sentence tokenization of Tulu documents is quite straightforward as it follows specific rules similar to other Indian languages and is carried out using IndicNLP[16] library [50]. After sentence segmentation, the number of Tulu sentences amounts to 10,000. These are then manually translated to Kannada by native Tulu speakers. Tulu sentences collected from various resources are in Kannada script. For visualization purpose, these texts are converted into Tigalari - a Tulu native script using *padaTulu*[17] software.

4.2.2 Automatic Text Generation for Kannada-Tulu Parallel Corpus - *KT1*

KT1 parallel corpus consists of very less number of parallel sentences. Automatic Text Generation (ATG) helps to construct the dataset for under-resourced languages automatically by creating synthetic data. To perform ATG, a sample dataset (sampkatl) consisting of 100 Kan-Tul parallel sentences of simple day-to-day conversational sentences are collected from Tulu language learning blogs[18,19] to extract the NEs used for ATG. These sentences are manually checked and the sentences which strictly follow the Subject-Object-Verb (SOV) word order are considered for ATG. In addition, sentences are thoroughly checked to ensure that there are no duplications. During the collection of this sample dataset care is taken to ensure that the size of vocabulary is maximized after ATG and there should be minimum or no postprocessing after ATG. Table 6 shows the statistics of the *KT1* parallel corpus.

[15]https://tuludictionary.in/dictionary/cgi-bin/web/frame.html
[16]libraryhttps://github.com/anoopkunchukuttan/indic_nlp_library
[17]Pada-Tulu
[18]http://gunapalaalva.blogspot.com/p/tulu.html
[19]https://www.raveeshkumar.com/2009/05/learn-tulu-online-commonly-used-phrases.html

Table 6: Details of Train and Test set

	Train set	Test set
Before ATG	9,000	1,000
After ATG	29,000	1,000

Table 7: Data sources and the number of Kannada sentences considered for manual translation to Tulu

Data sources	# of sentences
Wikipedia	9,585
IndoWordnet	20,665
Mann ki baath	13,871
Bible	12,429
Samanantar	16,424

4.2.3 Limitations of *KT1* Corpus

The limitation of *KT1* corpus is its lack of generalization, which stems from the use of a small and slightly biased dataset. This restricted dataset limits the model's ability to learn diverse patterns and adapt to new or unseen data. As a result, the MT models trained using *KT1* corpus may perform well in controlled scenarios but struggle in real-world applications.

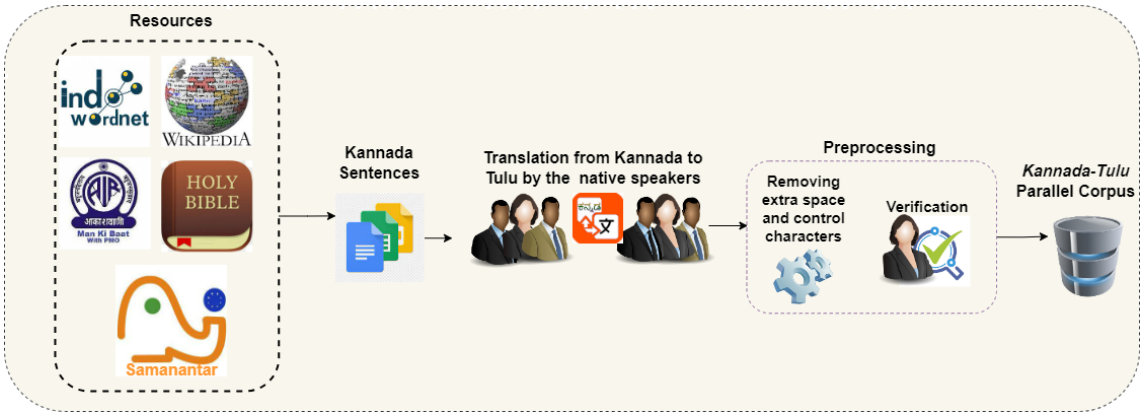


Figure 5: Framework of the Kan-Tul parallel corpus construction

4.2.4 Construction of Kannada-Tulu Parallel Corpus - *KT2*

To address the limitations of *KT1*, an extended version of the Kan-Tul corpus, *KT2*, has been constructed. The majority of this corpus is created through manual translation with the assistance of linguists and professional translators, ensuring greater accuracy and linguistic diversity. This expansion aims to enhance the quality and coverage of the Kan-Tul language

pair. The workflow for constructing the parallel corpus is shown in Figure 5 and the sources of Kannada sentences are shown in Table 7.

Table 8: Kannada and corresponding Tulu action verbs along with their English translations

Kannada verb	Tulu verb	English translation		Kannada verb	Tulu verb	English translation
ಹೋಗು	ಪೋ	go		ಎಸೆ	ದಕ್ಕ್	throw
ನಿಲ್ಲು	ಉಂತು	stand		ನಗು	ತೆಲಿಪು	laugh
ನೋಡು	ತೂಲ	see		ಹೇಳು	ಪನ್	tell
ತಿನ್ನು	ತಿನ್ನ್	eat		ಬಾ	ಬಲ್ಲ	come
ಮಾತನಾಡು	ಪಾತೇರ್	speak		ಜಿಗಿ	ಲಾಗು	jump
ಓದು	ಓದು	read		ತಗೆದುಕೊಳ್ಳು	ದೊತ್ತ್	take
ಓಡು	ಬಲಿಪು	run		ಹತ್ತು	ಮಿತ್ತಾರ್	climb
ಆಡು	ಗೊಬ್ಬು	play		ಮಾಡು	ಮಲ್ಪು	do
ಮಲಗು	ಜಿಪ್ಪು	sleep		ಕುಡಿ	ಪರ್	dring
ಹಿಡಿ	ಪತ್ತ್	catch		ಕೊಡು	ಕೊರು	give

Table 9: Sample tense suffixes and *vibhakti* suffixes for Kannada and Tulu languages

Kannada tense suffixes	Tulu tense suffixes	Cases	Kannada	Tulu
1:['ತ್ತೇನೆ','ತ್ತಿದ್ದೇನೆ','ವೆನು','ತ್ತೇನಾ','ತ್ತಿದ್ದೇನಾ','ವೆನಾ','ವುದಿಲ್ಲ','ವೆ','ವೆನೇನೊ'], 2:['ತ್ತಾನೆ','ತ್ತಿದ್ದಾನೆ','ತ್ತಾನಾ','ತ್ತಿದ್ದಾನಾ','ವನು','ವನಾ','ವುದಿಲ್ಲ','ವನೇನೊ'],3: ['ತ್ತಾಳೆ','ತ್ತಿದ್ದಾಳೆ','ತ್ತಾಳಾ','ತ್ತಿದ್ದಾಳಾ','ವುದಿಲ್ಲ','ವಳು','ವಳಾ','ವಳೇನೊ'], 4:['ತ್ತಾರೆ','ತ್ತಿದ್ದಾರೆ','ತ್ತಾರಾ','ತ್ತಿದ್ದಾರಾ','ವರು','ವರಾ','ವರೇನೊ','ವುದಿಲ್ಲ']	1:['ಪೆ','ವೊಂದುಲ್ಲೆ','ಪೆ','ಪೆನಾ','ವೊಂದುಲ್ಲೆನಾ','ಪೆನಾ','ಪುಜಿ','ಪೆ','ಪೆನ ದಾನೆ'], 2:['ಪೆ','ವೊಂದುಲ್ಲೆ','ಪೆನಾ','ವೊಂದುಲೆನಾ','ಪೆ','ಪೆನಾ','ಪುಜೆ','ಪೆನ ದಾನೆ'], 3: ['ಪಲ್','ವೊಂದುಲ್ಲಲ್','ಪಲಾ','ವೊಂದುಲ್ಲಲಾ','ಪುಜಲ್','ವಲ್','ವಲಾ','ವಲಾ ದಾನೆ'], 4: ['ಪೇರ್', 'ವೊಂದುಲ್ಲೇರ್', 'ಪೆರಾ', 'ವೊಂದುಲ್ಲೆರಾ' ,'ವೇರ್', 'ವೆರಾ', 'ಪೇರಾ ದಾನೆ', 'ಪುಜೆರ್']	Dative	['ಕ್ಕೆ','ಗೆ']	['ಕ್','ಗ್']
		Accusative	['ವನ್ನು','ಯನ್ನು']	['ನ್']
		Locative	['ದಲ್ಲಿ','ಯಲ್ಲಿ']	['ಡ್','ಟ್']
		Ablative	['ದಿಂದ','ಯಿಂದ']	['ಡ್']

4.2.5 Automatic Text Generation for Kannada-Tulu Parallel Corpus - *KT2*

ATG for *KT2* utilizes the rule-based NLG for ATG (rule-based ATG) which automatically generates the text according to the specified linguistic rules. This rule-based ATG creates new text rather than duplicating or adding slight linguistic variations to the existing data. The proposed rule-based ATG involves generating text by randomly selecting subjects, objects and verbs from the list of subjects, objects, and verbs respectively. These subjects, objects and verbs which are used in day-to-day conversation are collected from the available online resources. The steps involved in developing the proposed rule-based ATG system are given below:

1. 20 Kannada and corresponding Tulu action verbs are randomly selected and these verbs along with their English translations are shown in Table 8.

Table 10: Sample subject, object, and verb combinations along with rule sets for Kannada and Tulu languages

Rule set	Subject	Object	Verb	Description
Kannada				

for all verbs:
if subject is 1st person singular
for suffix in list of suffixes:
get verbal_inflection
for all objects in list of vibhakti suffixes:
if object belongs to dative case and contain noun indicating place:
get object_inflection
return subject+object_inflection+verbal_inflection
	ನಾನು	ಪೇಟೆಗೆ	ಹೋಗುತ್ತೇನೆ	ನಾನು = '1st person singular'
				ಪೇಟೆಗೆ = 'noun dative case'
				ಹೋಗುತ್ತೇನೆ = 'present tense singular'

Tulu				

for all verbs:
if subject is 3rd person singular masculine
for suffix in list of suffixes:
get verbal_inflection
for all objects in list of vibhakti suffixes:
if object belongs locative case and contain noun indicating place:
get object_inflection
return subject+object_inflection+verbal_inflection
	ರಾಮೆ	ಇಲ್ಲ್ ಡ್	ಉಂತುವನ ದಾನ	ರಾಮೆ = '3rd person singular masculine'
				ಇಲ್ಲ್ ಡ್ = 'noun locative case'
				ಉಂತುವೆ = 'future tense singular'

2. 36 distinct suffixes belonging to present and future tenses in both Kannada and Tulu languages are selected to inflect the verbs and the samples of such suffixes are shown in Table 9.

3. Based on day-to-day conversation, 40 objects each for Kannada and Tulu languages which are compatible with the 20 action verbs mentioned in Table 9 are selected.

4. 4 different cases (*vibhakti*) are selected for object inflections for both Kannada and Tulu languages and these cases are shown in Table 9.

5. 10 sample words belonging to masculine and feminine genders and 4 pronouns (both including singular and plural) are randomly selected from the online resources.

6. Based on *vibhakti*/cases, PNG, tense, the intent of the verb and complexity of the verbal inflection, 14 categories are created for Kannada and 24 categories are created for Tulu as most of the words in Tulu change their spelling when a verb in its base form is inflected with the suffix/es.

7. A handcrafted implementation is carried out based on the rules set to get the unique combinations of subjects, objects, and verbs in Subject-Object-Verb (SOV) order. The sample rule sets along with sample subject, object, and verb combinations, for both Kannada and Tulu languages are shown in Table 10.

8. The order of subjects, objects, and verbs in unique SOV combinations is changed into OSV in order to get syntactical variation in the generated text.

With this arrangement, the proposed rule-based ATG with a set of 14 rules for Kannada and 24 rules for Tulu resulted in 57,600 unique Kannada-Tulu parallel sentences. However, as both Kannada and Tulu are morphologically rich languages, it is very difficult to define an exhaustive set of linguistic rules.

Table 11: Annotation guidelines along with sample Kannada and Tulu sentences followed by their English translations

Label	Guideline	Sample Kannada sentence	Sample Tulu sentence	English translation	Description
Correct	Sentence with correct syntax and semantic information is tagged as 'correct'	ಅವರು ಮನೆಗೆ ಹೋಗುವುದಿಲ್ಲ	ಅರ್ ಇಲ್ಲ್ ಗ್ ಪೋಪುಜಿ ರ್	They are not going to home	The sentence conveys a complete information and is syntactically and semantically correct
Incorrect	Sentence with incorrect syntax and/or semantic information is tagged as 'incorrect'	ಮನೆದಲ್ಲಿ ಅವಳು ಹೋಗುತಾಳಾ	ಇಲ್ಲ್ ಟ್ ಆಲ್ ಪೋಪಲಾ	Does she go to home	In sample Kannada sentence, the word 'ಮನೆದಲ್ಲಿ' is syntactically wrong and in sample Tulu sentence, though the word 'ಇಲ್ಲ್ ಟ್' is syntactically correct, it used in a wrong context which is semantically incorrect

Table 12: Sample wrong patterns in Kannada and Tulu sentences

Kannada	Tulu
['ಮಾರುಕಟ್ಟೆದಲ್ಲಿ','ಮಾರುಕಟ್ಟೆ ಕ್ಕೆ','ಶಾಲೆದಲ್ಲಿ','ಮನೆಕ್ಕೆ','ಪೇಟೆ ಕ್ಕೆ','ಪೇಟೆದಲ್ಲಿ','ಹಳ್ಳಿದಲ್ಲಿ','ಹ ಳ್ಳಿಕ್ಕೆ','ಶಾಲೆಕ್ಕೆ','ಮನೆದಲ್ಲಿ' 'ಕಾರ್ಯಕ್ರಮಗೆ', 'ಟಿವಿದಲ್ಲಿ', 'ಟಿವಿಕ್ಕೆ ', 'ಚಿತ್ರಕ್ಕೆ' , 'ತಿಂಡಿವನ್ನು']	['ಇಲ್ಲ್ ಕ್','ಇಲ್ಲ್ ಟ್',' ಹಳ್ಳಿಕ್ ','ಶಾಲೆಕ್',' ಹಳ್ಳಿಟ್', 'ಪೇಟೆಕ್', 'ಪೇಟೆಟ್', 'ಎದ್ದೆ ಗ್', 'ಟಿವಿಡ್ದ್ ', 'ಟಿವಿಟ್ ', ಪರಂದ್ ಡ್', 'ಮೆಲ್ಲಟ್', 'ಜೋರಕ್ ', 'ಜೋರಡ್', 'ಗೊಬ್ಬುಗ್', 'ಬೇಲೆಟ್']

4.2.5.1 Evaluation of the Text Generated by ATG for *KT2*

Rule-based approaches can be useful for processing simple text and these approaches can be prone to errors, especially when dealing with morphologically rich and agglutinative languages due to rich verbal and object inflections [51]. The evaluation of the proposed ATG is carried out in two steps: Manual verification to find the wrong patterns in a selected set of sentences and automatic identification of these wrong patterns in the given set of sentences. The verification details are given below:

- Manual verification - of the generated text involves manually checking whether the generated sentences are syntactically and semantically correct or not. These sentences are then annotated with 'correct' or 'incorrect' labels depending on whether the sentence is correct or incorrect respectively. This annotation process involves 6 native speakers

Table 13: Statistics of Kannada-Tulu parallel corpus and details about Train and Test set

Languages	# of words	# of unique words	Average sentence length
Kannada	7,04,937	1,36,129	6
Tulu	7,81,603	1,33,568	7
Kan-Tul parallel sentences	**Train set**	**Test set**	
	1,14,762	2,500	
	Total	**1,17,262**	

Table 14: Performance measure of the MT models for *KT1* parallel corpus

Before ATG				
	BLEU		**CHRF**	
Models	**Kan-Tul**	**Tul-Kan**	**Kan-Tul**	**Tul-Kan**
SMT	10.56	10.06	19.56	18.03
RNN	20.59	18.19	25.68	23.81
BiRNN	21.33	20.45	25.80	22.66
Transformer	22.89	22.03	26.92	25.40
Transformer+BPE	**23.06**	**22.80**	28.01	24.90
After ATG				
SMT	15.83	14.98	20.55	19.36
RNN	31.73	30.47	43.05	38.65
BiRNN	39.97	38.88	48.60	47.51
Transformer	41.02	40.23	49.75	47.39
Transformer+BPE	**41.82**	**40.94**	49.90	47.88

of the respective languages as annotators; 3 for each language. Guidelines provided to the annotators to carry out the annotations along with the sample Kannada and Tulu sentences followed by their English translations are shown in Table 11. Words that violate the rules in the incorrect sentences (both in Kannada and Tulu) are identified and listed, for creating wrong patterns and a sample of such wrong patterns are shown in Table 12. This process ensures the quality and accuracy of the generated content. Out of 57,600 Kannada-Tulu parallel sentences, 12,000 sentences containing all the verbal and object inflections are selected and these sentences are used for the manual verification so that all possible wrong patterns can be captured.

- Automatic evaluation - of the sentences is carried out by matching the wrong patterns (obtained during manual verification) automatically. Out of 57,600 sentences generated, 45,600 sentences are considered for automatic evaluation. If a sentence consists of wrong patterns, such a sentence is annotated as 'incorrect' otherwise it is annotated as 'correct'.

Out of 57,600 parallel sentences generated by ATG, 44,288 parallel sentences are found to be correct after evaluation. These sentence pairs/parallel sentences combined with the

Table 15: Performance of the baselines for *KT2* parallel corpus after employing ATG

Models	Kan-Tul		Tul-Kan	
	BLEU	**CHRF**	**BLEU**	**CHRF**
RNN+GRU (before ATG)	07.33	15.02	08.50	17.00
RNN+GRU	09.86	19.60	10.00	20.00
RNN+LSTM	20.18	46.84	27.35	54.30
BiRNN+GRU	08.26	19.60	08.90	19.97
BiRNN+LSTM	**20.65**	47.28	**29.16**	54.92
Transformer+GRU	14.85	22.43	18.62	23.59
Transformer+LSTM	**23.87**	48.30	**30.16**	56.21
Transformer+GRU+BPE	20.50	25.70	21.66	31.12
Transformer+LSTM+BPE	**24.10**	50.29	**34.15**	59.80

72,974 manually translated sentence pairs/parallel sentences amounting to 1,17,262 sentence pairs/parallel sentences form the Kannada-Tulu parallel corpus and Table 13 shows the statistics of this corpus.

4.2.6 Kannada-Tulu Machine Translation Baselines

A set of encoder-decoder-based NMT approaches, the same models which are used for *KanSan* corpus, including RNN, BiRNN, and Transformer models, are trained using combinations of GRU and LSTM units: RNN+GRU, RNN+LSTM, BiRNN+GRU, BiRNN+LSTM, Transformer+GRU, and Transformer+LSTM. These models are experimented with to establish benchmarks for the Kannada-Tulu parallel corpus. Additionally, BPE tokenization is incorporated, as it has shown promising results in similar research.

4.2.7 Experiments and Results

This section presents the experiments and results for the *KT1* and *KT2* parallel corpora, both before and after TA. The findings highlight the impact of TA on model performance across these datasets. NMT models for translating Kan-Tul sentence pairs and vice versa are implemented using the open-source NMT framework, OpenNMT-py, similar to the Kan-San MT models with same configuration. Furthermore, evaluation metrics which are used to evaluate the Kan-San MT models (as mentioned in Section 4.1.4) are used to asses the performance of Kan-Tul MT models. Performance measures of the MT models for *KT1* corpus are shown in Table 14. From Table 14, it is clear that the performance of all the models are better after ATG. The performance of the proposed models are shown in Table 15 for *KT2* parallel corpus. The results illustrate that all the proposed NMT models exhibited considerably good BLEU and CHRF scores for both Kan-Tul and Tul-Kan translations. From Table 15, it is clear that models with GRU exhibited the lowest scores due to the simpler architecture of GRU which fails to capture the correct context of the words with complex structure. On the other hand, models with LSTM performed better, as LSTM architecture

captures long-term dependencies. LSTM architectures are beneficial for dealing with complex data, which makes them suitable for handling morphologically rich agglutinative languages like Kannada and Tulu.

4.3 Syllable-level Morpheme Segmentation and Boundary Feature Integration for Kannada–Tulu Neural Machine Translation

Morphemes are the smallest meaningful units of a language that provides functional relationship with words in a linguistic system [52]. In Linguistics, morphology deals with the study of the combination of morphemes that forms words and the segmentation of words into their constituent components or morphological forms, viz. stem and affix morphemes. Splitting the word or compound word into their constituent components is known as Morphological Segmentation (MS) [53]. Morpheme Boundary Detection (MBD) detects the boundary between a stem morpheme and its affix morpheme(s) in a given word and Stemming is the process of reducing a word into its stem morpheme by removing the affix morpheme(s) [54]. MBD captures linguistic features and that can be used in many NLP applications like MT. However, for many under-resourced languages, there is a lack of available datasets to effectively perform this task, limiting the progress in improving translation quality for these languages. In view of this, Kannada and Tulu annotated corpora for MBD and MS are constructed. The steps involved in constructing MBD corpora for Kannada and Tulu are given below:

- **Corpus Construction -** To construct Kannada dataset for MBD, 47,407 unique Kannada words are taken from IndoWordnet[20], out of which 38,707 are nouns and 8,700 are verbs. These words have simple morphological structure with only one suffix and no prefix(es). The noun list is free of noise, whereas the verb list contains compound verbs - a verb joined with another verb with a hyphen in between as shown in Table 16. These compound verbs are reduced into base forms based on their semantic information. Some of the compound verbs are manually replaced by the verbal form of the first verb in the compound verb (e.g.1 and e.g.3 in Table 16). Using handcrafted codes, few compound verbs are reduced into base forms either by removing the verb which is present after the hyphen or by removing only the hyphen and combining both the verbs (e.g.4 in Table 16). The resultant verbs in the base form are used for annotation.

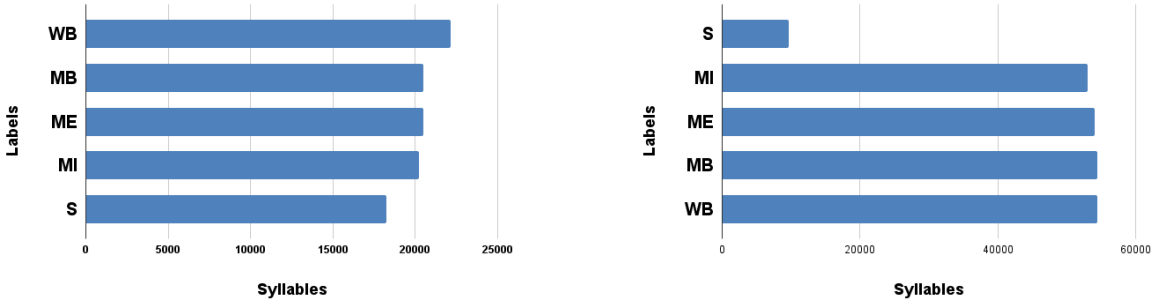
  Poovari is a monthly Tulu magazine that consists of Tulu articles in Kannada script in PDF form. Tulu text in PDF form is transformed into processable form using NewOCR[21] - a publicly available character recognizer that converts the text in PDF form to a string. These strings are tokenized into sentences and sentences are further

[20] https://www.cfilt.iitb.ac.in/indowordnet/
[21] https://www.newocr.com/

Table 16: Sample Kannada compound words, words after removing the duplicate verb and their English translations

Sl. No	Compound words	Words after removing duplicate verb	English translation
e.g.1	ನೃತ್ಯ-ಮಾಡು	ನರ್ತಿಸು	Dance
e.g. 2	ಹಾಡು-ಹೇಳು	ಹಾಡು	Sing
e.g. 3	ಚಿಂತೆ-ಮಾಡು	ಚಿಂತಿಸು	Think
e.g. 4	ಕೆಲಸ-ಮಾಡು	ಕೆಲಸಮಾಡು	Work

tokenized into words using the tokenizers provided by indicNLP[22] library. During pre-processing, digits, punctuation, and control characters are removed resulting in 20,482 unique words. Tulu words considered in this dataset consists of a suffix of length one only.



(a) Tulu MBD dataset

(b) Kannada MBD dataset

Figure 6: Distribution of labels in the Kannada and Tulu datasets

- **Data Annotation for MBD -** Preprocessed Kannada and Tulu words are syllabalized using syllabalizer[23]. The sequence of syllables of each word is then annotated by the respective Kannada and Tulu native speakers according to the following guidelines:
  - **Morpheme Begin (MB)** - the syllable at the beginning of the stem morpheme in the word is labeled as MB
  - **Morpheme End (ME)** - the syllable at the end of the stem morpheme indicating the morpheme boundary for stem morpheme is labelled as ME
  - **Morpheme Intermediate (MI)** - the intermediate syllables are labeled as MI
  - **Suffix (S)** - the syllable that indicates the suffix morpheme of the word is labeled as S

[22] https://github.com/anoopkunchukuttan/indic_nlp_library
[23] https://github.com/libindic/syllabalizer

– '.' - indicates Morpheme Boundary (MB) between words in the input sequence

All the annotations are manually verified to avoid wrong tagging. To ensure the quality of annotation, several words are annotated prior to the annotation of actual data. After annotation, the annotated corpora contains 2,77,236 Kannada syllables and 1,01,603 Tulu syllables with their corresponding tags including word boundaries. Figure 6 shows the label distribution of the annotated Kannada and Tulu datasets and the sample dataset after annotation is shown in Table 17.

Table 17: Sample annotated Kannada and Tulu corpora

Kannada		Tulu	
Syllable	**Tag**	**Syllable**	**Tag**
ಸ್ಟೂ	MB	ಜೋ	MB
ಲ	MI	ಕು	MI
ವಾ	ME	ಲೆ	ME
ದ	S	ಗ್	S
.	.	.	.
ಮ	MB	ವ	MB
ಡಿ	MI	ನ	MI
ಲು	ME	ಸ್	ME
.	.	.	.

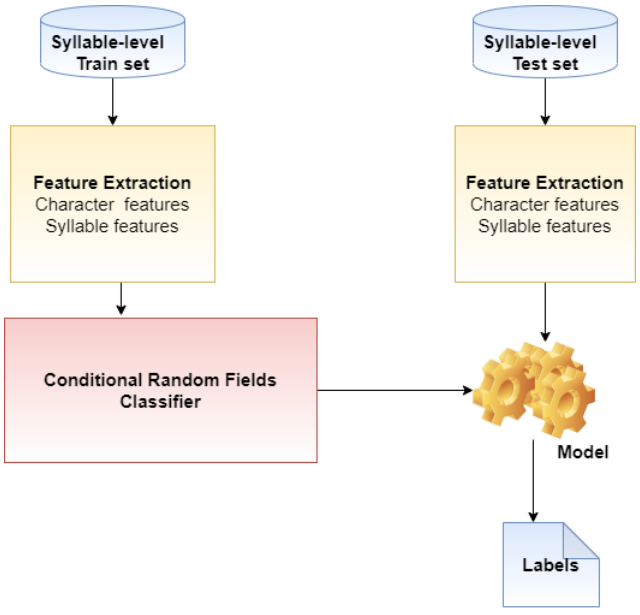


Figure 7: Framework of the proposed methodology

Table 18: Hyperparameters and their values used in the the proposed work

Model	Hyperparameters	Values
CRF	algorithm	lbfgs
	c1	0.065
	c2	0.002
	max_iterations	200
	all_possible_transitions	True

Table 19: Features used in the proposed CRF model

Syllable features		Character features	
A word	Previous syllable -1	A word	Previous syllable -1
Length of syllable	Previous syllable -2	Length of character	Previous syllable -2
Begining of word	Previous syllable -3	Begining of word	Previous syllable -3
End of Word	Next syllable +1	End of Word	Next syllable +1
A syllable	Next syllable +2	A character	Next syllable +2
	Next syllable +3		Next syllable +3

4.3.1 Proposed Method for Morpheme Boundary Detection

MS of Kannada and Tulu words at syllable-level is a sequence-labeling task and the methodology comprises of feature extraction followed by the model construction and evaluation. Framework of the proposed model is shown in Figure 7. It may be noted that, as the manually annotated corpora are used to train the model, necessary pre-processing is carried out during the corpus construction itself.

4.3.1.1 Feature Extraction

Feature extraction is the most essential as well as a critical task in building learning models and the performance of the learning models relies heavily on the extracted features. The relevant features extracted from the labeled data are used to train the learning models. Syllable-level and character-level features are more helpful as these features capture the insights of the words irrespective of whether the word is in the Train set or not [55]. Further, syllable-level and character-level features are found more beneficial in MS of words belonging to morphologically rich languages.

Table 20: Results of CRF model for MS of Kannada and Tulu words

Language	Features	Precision	Recall	F1 score
Kannada	Character features	0.95	0.94	0.95
	Syllable features	0.96	0.95	0.97
Tulu	Character features	0.84	0.84	0.84
	Syllable features	0.86	0.84	0.86

4.3.1.2 Model Construction

Conditional Random Fields (CRF) model - a kind of Machine Learning (ML) are fount beneficial in addressing sequence-labeling tasks [56]. Therefore, CRF models are trained individually using syllable-level and character-level features of Kannada and Tulu words to perform MS of Kannada and Tulu words. To achieve this, CRFSuite[24] a python tool is used which is widely used in Sequence Labeling tasks [57]. A linear-chain CRFSuite is the most popular type of CRFSuite class for tagging and in CRFSuite training, weight values are estimated. Higher weight values indicate that the relationship between observations and tags has been supported by the features. A trained CRFSuite labels the input by finding the label sequence that maximizes the conditional probability.

4.3.2 Experiments and Results for Morpheme Boundary Detection

The commonly used metrics to evaluate the performance of the classification models include precision, recall, and F1 scores along with macro and weighted averages. These evaluation methods differ based on how they assess the impact of samples or classes in the final score. Macro-average methods treat all classes equally, whereas weighted-average method calculates the scores taking into account the size of each class considering the number of samples. Hence, weighted average of precision, recall, and F1 score are used to evaluate the performance of the learning models. sklearn-crfsuite, a scikit-learn compatible Python-based CRFSuite wrapper is used to implement MS of Kannada and Tulu words. The hyperparameters and their values used to configure CRFSuite are shown in Table 18 and the list of syllable and character features used to train the CRF models are shown in Table 19.

A 10-Fold cross-validation is applied for both Kannada and Tulu datasets to check the effectiveness of the CRF models. The results obtained by the CRF models for MS of Tulu and Kannada words are shown in Table 20. CRF models trained using syllable features obtained F1 scores of 0.97 and 0.86 for Kannada and Tulu datasets, respectively. Using character features, CRF models obtained F1 scores of 0.95 and 0.84 for Kannada and Tulu datasets, respectively. CRF models trained using syllable features have exhibited slightly improved F1 scores for MS of Kannada and Tulu words as individual syllable efficiently represents a meaningful element of the input words better than characters. Further, character features fail to represent the meaningful units of words as the dependent components/vowels cannot stand alone when the Kannada and Tulu words are represented in their native script.

[24] https://sklearn-CRFSuite.readthedocs.io/en/latest/

Table 21: Performance of NMT models before and after integrating MB features

Language pair	**Before incorporating MB features**		**After incorporating MB features (@source)**		**After incorporating MB features (@ both source and target) (Proposed model)**	
	BLEU	CHRF	BLEU	CHRF	BLEU	CHRF
Kan-Tul	24.10	50.29	24.35	51.96	**24.57**	**52.03**
Tul-Kan	**34.15**	**59.80**	30.02	49.33	30.82	49.50

4.3.3 Proposed Method for Integrating Morpheme Boundary Features to Neural Machine Translation

Integrating MB features into NMT models can significantly enhance their performance, especially in languages with complex morphological structures or those that do not use spaces between words. Word boundary features help the model better understand the segmentation of text, which is crucial for accurate word alignment and translation. For languages with agglutinative characteristics, such as Kannada, Tamil, Tulu, or Sanskrit, this feature allows the model to distinguish word boundaries more effectively, improving both the fluency and accuracy of the translation. By incorporating such features, NMT systems can better handle tasks like word segmentation, enabling the model to learn more meaningful representations of the input text and produce higher-quality translations. This approach has shown promising results in both under-resourced and high-resource language pairs, contributing to improved handling of diverse linguistic structures [58].

In recent advancements in NMT systems, the incorporation of MB features has shown promising improvements. When these features are used at the source text level, the transformer models demonstrate better translation quality compared to those that do not include them [59]. A tagged corpus that identifies MB provides valuable context, allowing the model to better understand token segmentation and enhance translation accuracy. Motivated by the potential of enhancing translation accuracy, this work incorporates MB features to capture linguistic information from the source text, which is particularly useful for translating languages with complex word structures. Building on this, the MB features are extended to both the source and target texts, allowing the model to better align translations between languages. By providing additional context through MB tagging, the NMT system can improve its understanding of token segmentation and word relationships. This extended approach has shown to significantly enhance the fluency and precision of the generated translations. As a result, NMT models that leverage MB features are increasingly effective in producing more accurate and natural-sounding translations.

4.3.4 Experiments and Results of Morpheme Boundary Integration

Inspired by Supriya et al., [59] OpenNMT-py tool is used to implement Kan-Tul MT system along with the MB tagged *KT2* parallel corpora. The hyperparameters and their values are also used as per experiments carried out by Supriya et al., [59]. Table 21 shows the performance of NMT models before and after incorporating linguistic features.

4.4 Development of Part-Of-Speech Tagger and Integration of Part-Of-Speech Features for Neural Machine Translation

Parts-Of-Speech (POS) tagging is a preliminary step for various NLP tasks, such as MT, IR, and SA. By identifying the grammatical categories of words in a text, POS tagging helps to understand the context and relationships, thereby improving the accuracy and relevance of NLP tasks. However, for under-resourced language like Tulu lacks the availability of POS tagger dataset that can reduce the chance of integrating POS features to the NLP tasks involved in Tulu. To bridge this gap a benchmarked Tulu POS tagging dataset is constructed. To build the Tulu POS tagger dataset, 15,835 Tulu articles were collected from 'Times of Kudla'[25] - a newspaper that publishes both local and global news in Tulu language, using customized web scraper.

4.4.1 Pre-processing

Cleaning and preparing data for annotation is a crucial step in creating a high-quality annotated corpus, as it ensures accuracy and accelerates the annotation process. Sentence tokenization is applied to split Tulu news articles into individual sentences in order to structure the data for annotation. Further, removing duplicate sentences, URLs, and short sentences (sentences having less than 4 words) resulted in 15,000 sentences. Each sentence is then tokenized into words, which form the basis for assigning POS tags. This systematic pre-processing resulted in 1,99,787 words with minimum 4 words, maximum 43 tokens, and 12 words on the average in a sentence.

4.4.2 Annotation

The annotation scheme for Tulu POS tagger is based on the Universal POS tags[26], ensuring consistency with widely recognized standards. Each word in the given sequence

[25]https://www.timesofkudla.com/
[26]https://universaldependencies.org/u/pos/

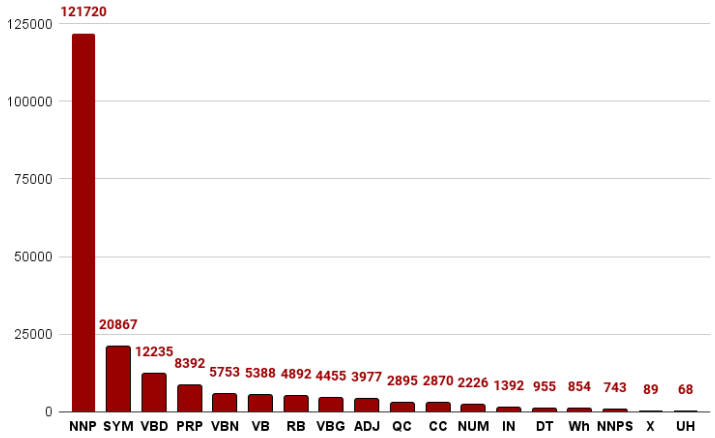


Figure 8: Classwise distribution of Tulu POS tagger dataset

is annotated by three annotators following guidelines provided by the Universal POS tags. These guidelines outline the correct application of tags to words, ensuring uniformity across annotations and improving the overall quality of the dataset. Further, multiple annotators help to enhance the reliability of the tagged corpus. As per Universal POS tags, the proposed Tulu POS tagger corpus contains 18 tags.

4.4.3 Corpus Statistics

Tulu POS tagger dataset consists of 199,787 tokens collected from 15,000 articles, categorized into 18 groups: NNP, SYM, VBD, PRP, VBN, VB, RB, VBG, ADJ, QC, CC, NUM, IN, DT, Wh, NNPS, X, and UH. Figure 8 shows the glimpse of statistics of the dataset and also indicates that the dataset is imbalanced. The most frequent category is 'NNP', reflecting the common use of nouns in articles, whereas the second most frequent categories are 'SYM' and 'VBD'. Symbols, including punctuation, are vital in news articles for readability, while past tense verbs dominate, as news tends to describe past events.

4.4.4 PosT: Proposed Method for Building Tulu Parts-Of-Speech Tagger

POS tagging is a linguistic activity and a fundamental task in NLP, where each word in a sentence is labeled with its grammatical category, such as noun, verb, or adjective. In this work, a wide range of ML (PosT_ML and PosT_CRF), DL (PosT_DL) and TL (PosT_TL) are implemented using the proposed Tulu dataset for POS tagging. To illustrate the efficacy of the proposed baselines, the baselines are also evaluated on

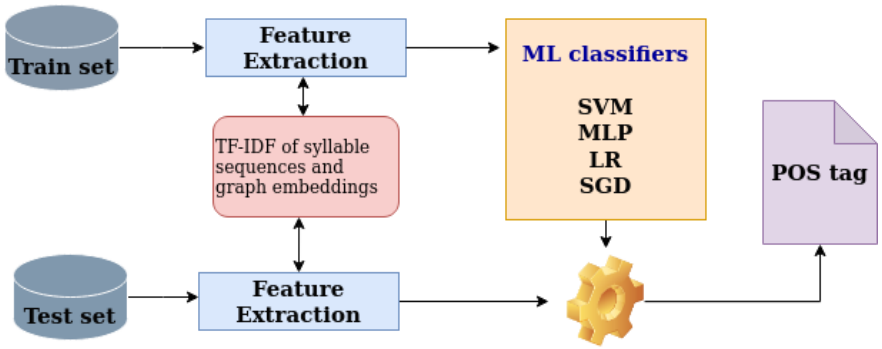


Figure 9: Framework of PosT_ML model

Table 22: Features used in the proposed PosT_CRF model and its hyperparameters and their values

CRF Features		Hyperparameters	Values
A word	Next word	algorithm	lbfgs
Length of word	Prefix of length 2	c1	0.065
Beginning of sentence	Prefix of length 3	c2	0.002
End of sentence	Prefix of length 4	max_iterations	200
Is current word digit	Suffix of length 2	all_possible _transitions	True
Is current word punctuation	Suffix of length 3	-	
Previous word	Suffix of length 4	-	

publicly available Kannada[27] and Sanskrit[28] POS tagger datasets.

POS is a sequence labeling problem where each and every component of a sequence is labeled with one of the predefined tags. To accomplish this, four models: PosT_ML, PosT_CRF, PosT_DL, and PosT_TL, are proposed. After conducting a wide range of empirical studies, the proposed models employ the best hyperparameters and their values for optimal POS tagging solutions. Descriptions of these models are given in following subsections.

4.4.4.1 PosT_ML

The proposed PosT_ML model implements traditional ML classifiers (Support Vector Machine (SVM), Multilayer Perceptron (MLP), Logistic Regression (LR), and Stochastic Gradient Descent (SGD)) for POS tagging and consists of two modules: Feature extraction and Model construction, and the framework of the model is depicted in Figure 9.

[27]https://ltrc.iiit.ac.in/showfile.php?filename=downloads/kolhi/

[28]https://github.com/premjithb/Part-of-Speech-Tagging-Machine-Learning-Deep-Learning-Word2vec-fasttext/tree/master

Feature Extraction

The objective of feature extraction is to extract distinguishing features from the given words to build POS tagger and the feature extraction techniques used in this paper are as follows:

- **Term Frequency- Inverse Document Frequency (TF-IDF) of Syllable Sequences** - Syllables are a unit of pronunciation having one vowel sound and for Indian languages in native script, syllable representation is more meaningful. The syllable sequences or n-grams are the sequences of syllables that represent syllable features and in this work the range of n = (1, 3). libindic[29] library is used for syllabification of words in the given dataset and the extracted syllable sequences are converted to TF-IDF vectors using TfidfVectorizer[30]. TF-IDF vectors provide the normalized representation of text documents by mitigating the influence of excessively repeated words.
- **Graph Embeddings** - Graphs are commonly used data structures that represent relationships between elements through vertices and edges and are used to capture the dependencies between words in sequences. DL methods can transform these graphs into vectors of real numbers with specific dimension [60]. In this work, graph embeddings are generated using a Python library node2vec[31], which takes a graph as input and produces feature vectors for each vertex considering edge as relationship between two vertices. A customized node2vec model is implemented using cleaned Tulu/Kannada/Sanskrit Wikipedia text tokenized into sentences. The model is trained with parameters set to dimension = 64, walk_length = 10, num_walks = 200, and workers = 4.

Classifier Construction

The proposed methodology builds the ML classifiers: SVM, MLP, LR, and SGD, using the TF-IDF of syllable sequences and graph embeddings each to determine the POS tag of an unlabeled word.

4.4.4.2 PosT_CRF

CRF algorithm performs well in modeling the conditional probability distribution of tags, like POS, for sequences of observations like words in a sentence [57]. The proposed

[29]https://github.com/libindic/syllabalizer
[30]https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
[31]https://pypi.org/project/node2vec/

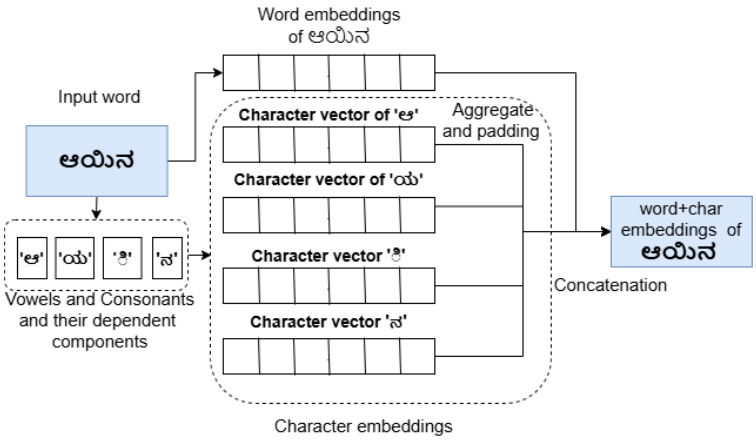


Figure 10: Word+char embeddings

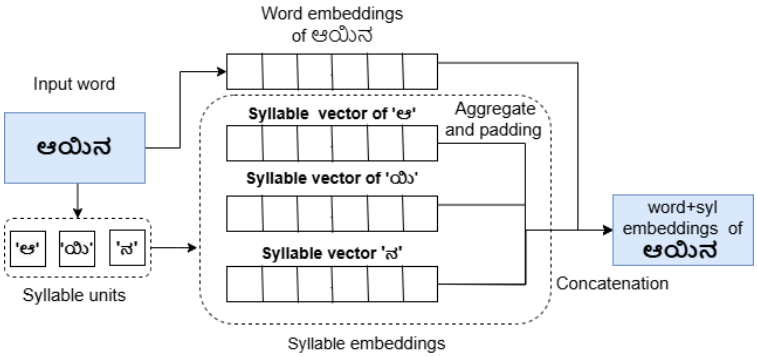


Figure 11: Word+syl embeddings

PosT_CRF model utilizes text features such as word length, previous words, and next words and the syllable features to enhance the performance of the learning model. By leveraging syllable information, the model gains insights into the structure of words, particularly through the extraction of prefix and suffix, contributing to the overall performance. The proposed CRF classifier is implemented using sklearn_crfsuite library, which serves as a wrapper for CRF implementation. Text features and hyperparameters and their values used in the proposed PosT_CRF model are shown in Table 22.

4.4.4.3 PosT_DL

DL models have shown considerable performance over ML models in tagging POS to each token in the given sequence. The proposed PosT_DL consists of implementing sequential BiLSTM models trained individually with Keras embeddings, fastText word embeddings, Byte Pair Embeddings (BPE) embeddings, word+character (word+char) and word+syllable (word+syl) embeddings, and Sequence-to-Sequence (Seq2Seq) BiLSTM model trained with Keras embeddings. These models have two modules: Feature

Table 23: Hyperparameters and their values used to train the BiLSTM models

Hyperparameters	Values
Epochs	10
Batch size	64
Learning rate	0.001
Optimizer	Adam
Loss fuction	sparse_categorical_ crossentropy
Drop out	0.01
Early stopping	True
Verbose	1

extraction and Classifier construction and the description of these models follows:

Feature Extraction

The feature extraction techniques utilized in Post_DL are described below:

- **Keras embeddings**[32] - Neural networks usually begin with an embedding layer as the first layer, which maps input data from a high-dimensional space to a lower-dimensional one. This allows the network to capture relationships between inputs and process the data more efficiently.
- **fastText**[33] **Word Embeddings** - is a pre-trained word embeddings developed by Facebook AI Research (FAIR). It is a powerful open-source library designed for both word representation and classification, which includes pre-trained models for more than 157 languages in their native scripts, each trained on diverse textual data sources, including Wikipedia. The work incorporated in this thesis, utilizes Tulu/Kannada/Sanskrit fastText pre-trained word embeddings to represent Tulu/Kannada/Sanskrit words in native script.
- **BPE**[34] **sub-word embeddings** - are pre-trained sub-word embeddings, where sub-words represent substrings of a word, capturing finer linguistic details. Using BPEmb library, words are split into sub-words for more granular representation. Similar to word vectors, a sub-word2vec model of size 100 is trained on these sub-words using the Skipgram approach, which helps in learning relationships at the sub-word level.
- **word+char and word+syl embeddings** - character embeddings encode words as sequences of individual characters, allowing them to capture morphological information and provide robustness to misspellings and variations. This approach

[32]https://www.tensorflow.org/api_docs/python/tf/keras/layers/Embedding
[33]https://fasttext.cc/docs/en/crawl-vectors.html
[34]https://bpemb.h-its.org/

generates continuous representations that effectively represent rare words and the structural properties of language. Character embeddings enhances the model's ability to understand and process text. In this work, character and syllable embeddings are generated using a Convolutional Neural Network (CNN) model, where character embeddings effectively capture consonants, dependent vowels, and combinations of multiple consonants as distinct units. In contrast, syllable embeddings provide representations for each syllable unit, enhancing the model's understanding of linguistic structure. Character embeddings are aggregated with word embeddings and syllable embeddings are aggregated with word embeddings, as shown in figures 10 and 11 respectively. Further, these embeddings are post-padded to create comprehensive representations. It may be noted that, character and syllable embedding dimensions considered are 30 and 50 respectively.

Classifier Construction

This section details the implementation of BiLSTM models, utilizing a sequential neural network architecture and Seq2Seq approach, allowing them to capture the information effectively by processing input data in both the directions for POS tagging.

- **BiLSTM model with sequential neural network** - BiLSTM refers to a sequence model that incorporates two LSTM layers: one processes the input in the forward direction, while the other in backward direction. This dual-direction approach enhances the model's ability to capture the dependencies within the data, allowing it to understand the relationships between elements in a sequence in a better way.
- **BiLSTM model with Seq2Seq architecture** - POS tagging is a sequence labeling task that is effectively modeled using a Seq2Seq framework. In this approach, a BiLSTM network produces a POS tag for each input word, as the lengths of the input and output sequences are identical. The architecture employs an encoder-decoder structure, where the entire sequence of words is represented as a single vector by the encoder. This vector is then passed to the decoder, which generates the corresponding POS tags for each input word.

Table 23 shows the hyperparameters and their values used to train the BiLSTM models.

4.4.4.4 Post_TL

Pre-trained language models have recently achieved state-of-the-art performance in a wide range of NLP tasks including POS tagging and the general framework of using

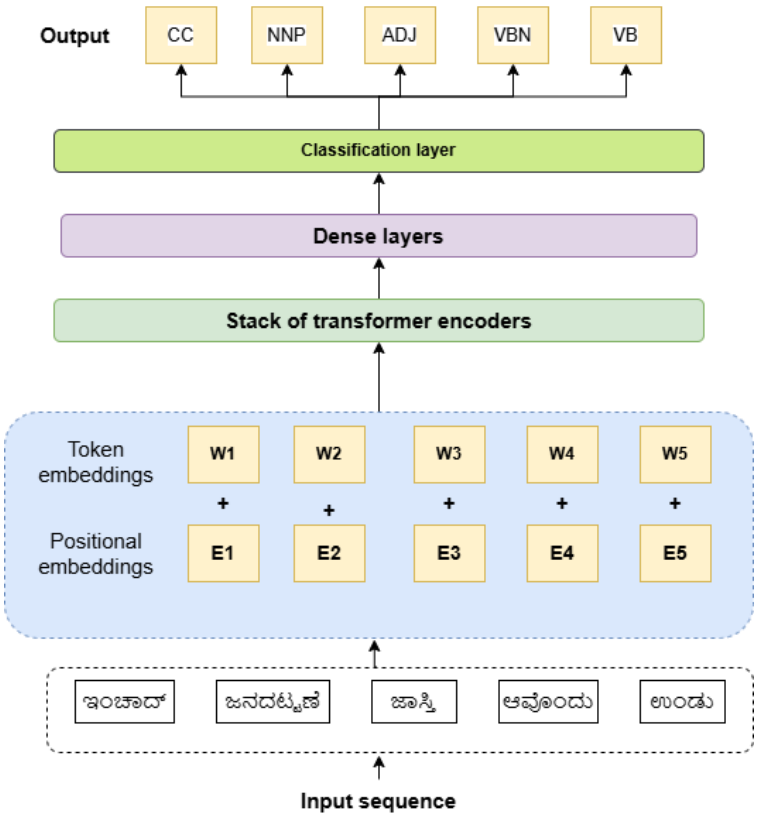


Figure 12: Framework of POS tagger using pre-trained transformer models

Table 24: Hyperparameters and their values used to build customized mBERT model for Tulu and for fine-tuning both customized mBERT and mBERT models

For building customized mBERT		**For fine-tuning**	
Hyperparameters	Values	Hyperparameters	Values
bTest	False	Batch size	16
vocab_size	10000	Learning rate	2.00E-05
verbose	0	Optimizer	adam
epochs	50	Epochs	20
layers	4	Drop out	0.001
Classifier_units	100	load best model at end	TRUE

Table 25: Performances of the baselines for Tulu POS tagging with Train-Test ratio 80:20

Features	Model	Precision	Recall	F1 score	Accuracy
Syllable sequence	CRF	0.81	0.82	0.81	0.82
word+char embeddings	BiLSTM	0.92	0.91	**0.90**	**0.91**
word+syl embeddings	BiLSTM	0.83	0.84	0.82	0.84
Graph embeddings	SVM	0.84	0.85	0.83	0.85
	MLP	0.82	0.83	0.81	0.83
	LR	0.67	0.69	0.63	0.69
	SGD	0.78	0.78	0.75	0.78
TF-IDF with syllable sequences	SVM	0.85	0.86	0.84	0.86
	MLP	0.85	0.71	0.76	0.71
	LR	0.84	0.84	0.83	0.84
	SGD	0.84	0.60	0.68	0.60
Kears embeddings	BiLSTM	0.84	0.84	0.82	0.84
fastText word embeddings	BiLSTM	0.86	0.60	0.67	0.60
BPE embeddings	BiLSTM	0.71	0.56	0.62	0.56
Keras embeddings	Seq2Seq BiLSTM	0.64	0.68	0.65	0.68
-	Customized mBERT	0.81	0.79	0.77	0.79

Table 26: Performances of the baselines for Tulu POS tagging with Train-Test ratio 90:10

Features	Model	Precision	Recall	F1 score	Accuracy
Syllable sequence	CRF	0.90	0.85	0.87	0.85
word+char embeddings	BiLSTM	0.94	0.95	**0.94**	**0.95**
word+syl embeddings	BiLSTM	0.93	0.92	0.91	0.93
Graph embeddings	SVM	0.87	0.88	0.87	0.87
	MLP	0.84	0.83	0.82	0.84
	LR	0.87	0.88	0.87	0.87
	SGD	0.84	0.86	0.84	0.84
TF-IDF with syllable sequences	SVM	0.87	0.87	0.86	0.87
	MLP	0.85	0.71	0.76	0.71
	LR	0.85	0.66	0.71	0.85
	SGD	0.77	0.60	0.66	0.77
Keras embeddings	BiLSTM	0.86	0.60	0.67	0.60
fastText word embeddings	BiLSTM	0.63	0.67	0.64	0.63
BPE embeddings	BiLSTM	0.71	0.56	0.62	0.56
Keras embeddings	Seq2Seq BiLSTM	0.64	0.68	0.65	0.68
-	Customized mBERT	0.81	0.79	0.77	0.79

Table 27: Performances of the baselines for Kannada POS tagging with Train-Test ratio 80:20

Features	Model	Precision	Recall	F1 score	Accuracy
Syllable sequence	CRF	0.84	0.80	0.81	0.80
word+char embeddings	BiLSTM	0.84	0.83	**0.84**	**0.83**
word+syl embeddings	BiLSTM	0.77	0.76	0.74	0.76
Graph embeddings	SVM	0.86	0.83	0.82	0.83
	MLP	0.72	0.70	0.68	0.70
	LR	0.82	0.80	0.80	0.80
	SGD	0.66	0.69	0.66	0.69
TF-IDF with syllable sequences	SVM	0.79	0.79	0.78	0.79
	MLP	0.62	0.61	0.60	0.61
	LR	0.72	0.66	0.65	0.66
	SGD	0.70	0.67	0.66	0.67
Keras embeddings	BiLSTM	0.74	0.72	0.69	0.72
fastText word embeddings	BiLSTM	0.74	0.73	0.71	0.73
BPE embeddings	BiLSTM	0.87	0.82	0.80	0.82
Keras embeddings	Seq2Seq BiLSTM	0.74	0.69	0.69	0.69
-	fine-tuned mBERT	0.85	0.81	0.82	0.81

Table 28: Performances of the baselines for Kannada POS tagging with Train-Test ratio 90:10

Features	Model	Precision	Recall	F1 score	Accuracy
Syllable sequence	CRF	0.90	0.87	0.88	0.90
word+char embeddings	BiLSTM	0.92	0.90	**0.91**	**0.93**
word+syl embeddings	BiLSTM	0.91	0.92	0.90	0.92
Graph embeddings	SVM	0.88	0.79	0.81	0.88
	MLP	0.86	0.76	0.79	0.86
	LR	0.85	0.77	0.79	0.85
	SGD	0.70	0.73	0.70	0.70
TF-IDF with syllable sequences	SVM	0.91	0.86	0.87	0.91
	MLP	0.79	0.69	0.73	0.79
	LR	0.84	0.76	0.78	0.84
	SGD	0.88	0.79	0.81	0.88
Keras embeddings	BiLSTM	0.82	0.65	0.71	0.82
fastText word embeddings	BiLSTM	0.74	0.73	0.71	0.73
BPE embeddings	BiLSTM	0.87	0.82	0.8	0.82
Keras embeddings	Seq2Seq BiLSTM	0.74	0.69	0.69	0.69
-	fine-tuned mBERT	0.85	0.81	0.82	0.81

Table 29: Performances of the baselines for Sanskrit POS tagging with Train-Test ratio 80:20

Features	Model	Precision	Recall	F1 score	Accuracy
Syllable sequence	CRF	0.87	0.81	0.84	0.87
word+char embeddings	BiLSTM	0.90	0.89	**0.89**	**0.90**
word+syl embeddings	BiLSTM	0.92	0.85	0.88	0.89
Graph embeddings	SVM	0.62	0.59	0.59	0.62
	MLP	0.60	0.56	0.56	0.77
	LR	0.62	0.54	0.54	0.65
	SGD	0.63	0.49	0.51	0.76
TF-IDF with syllable sequences	SVM	0.63	0.49	0.51	0.77
	MLP	0.72	0.63	0.65	0.80
	LR	0.57	0.44	0.45	0.73
	SGD	0.61	0.48	0.49	0.74
Kears embeddings	BiLSTM	0.79	0.75	0.76	0.75
fastText word embeddings	BiLSTM	0.71	0.65	0.62	0.65
BPE embeddings	BiLSTM	0.66	0.65	0.62	0.65
Keras embeddings	Seq2seq BiLSTM	0.71	0.72	0.71	0.72
-	sanskritBERT	0.18	0.37	0.22	0.37

Table 30: Performances of the baselines for Sanskrit POS tagging with Train-Test ratio 90:10

Features	Model	Precision	Recall	F1 score	Accuracy
Syllable sequence	CRF	0.87	0.87	0.87	0.90
word+char embeddings	BiLSTM	0.93	0.93	**0.92**	**0.93**
word+syl embeddings	BiLSTM	0.88	0.88	0.87	0.88
Graph embeddings	SVM	0.77	0.74	0.72	0.74
	MLP	0.81	0.8	0.79	0.8
	LR	0.66	0.66	0.62	0.66
	SGD	0.72	0.67	0.63	0.67
TF-IDF with syllable sequences	SVM	0.61	0.59	0.58	0.65
	MLP	0.60	0.57	0.56	0.74
	LR	0.61	0.47	0.47	0.63
	SGD	0.69	0.6	0.62	0.81
Kears embeddings	BiLSTM	0.79	0.78	0.78	0.78
fastText word embeddings	BiLSTM	0.70	0.63	0.58	0.63
BPE embeddings	BiLSTM	0.58	0.58	0.54	0.58
Keras embeddings	Seq2seq BiLSTM	0.65	0.65	0.64	0.65
-	sannskritBERT	0.16	0.34	0.20	0.34

pre-trained transformer models for POS tagging is depicted in Figure 12. In this study, a pre-trained mBERT model is further trained using Tulu Wikipedia text to include Tulu language along with other languages in mBERT model to use it for Tulu POS tagging. Tulu text utilized in this work is extracted from the Tulu Wikipedia[35] website using a custom Python script. After removing duplicate sentences, blank lines, and English content, the cleaned text consists of 1,88,509 sentences, which were then used to further train the mBERT model. mBERT model which is already proficient in multiple languages, is customized to handle Tulu by loading its tokenizer via 'AutoTokenizer' associated with the mBERT model. During this process, configuration files and the vocabulary specific to Tulu are saved, which are essential for building customized mBERT model. This enables the model to process Tulu text alongside the languages it was originally trained on. Hyperparameters and values used for fine-tuning the customized mBERT model and the original mBERT model are shown in Table 24.

4.4.5 Experiments and Results for Parts-Of-Speech Tagger

Several experiments are conducted by using different features and different classifiers using the newly created Tulu POS tagger dataset and the available Kannada and Sanskrit POS tagger datasets. Experiments are carried out on two Train and Test splits (80:20 and 90:10). Tables 25, 26, 27, 28, 29, and 30 shows performances of the proposed baselines on the two Train-Test ratios for Tulu, Kannada, and Sanskrit POS tagging datasets respectively. From the tables, it is clear that BiLSTM models trained with word+char embeddings outperformed the other models for both Tulu and Kannada POS tagging task with Train-Test ratio 90:10.

4.4.6 Proposed Method for Integrating Parts-Of-Speech Features to Neural Machine Translation

Integrating POS features into NMT models has been shown to significantly improve translation accuracy and fluency. POS tags provide valuable syntactic information, helping the model understand the grammatical structure of sentences, such as identifying verbs, nouns, adjectives, and other parts of speech. This knowledge allows the NMT system to better align words in the source and target languages, leading to more accurate and contextually appropriate translations. For example, POS tags can guide the model in selecting the correct word order or in handling ambiguities in the source language. Research has demonstrated that incorporating POS information into NMT models, either by adding it as an extra input feature or by using joint representations,

[35]https://tcy.wikipedia.org/wiki/

Table 31: Performance of NMT models before and after incorporating POS features

Language pair	Before incorporating POS features		After incorporating POS features (@source)		After incorporating POS features (@ both source and target)	
	BLEU	CHRF	BLEU	CHRF	BLEU	CHRF
Kan-Tul	24.10	50.29	30.32	54.69	**30.35**	**55.00**
Tul-Kan	34.15	59.80	39.66	62.08	**39.84**	**63.58**
Kan-San	09.84	26.37	16.13	30.50	**16.28**	**31.07**
San-Kan	12.63	30.80	13.04	34.71	**13.24**	**35.11**

leads to improved translation quality, particularly for languages with complex syntactic structures. This technique has shown particular promise for low-resource languages, where the lack of parallel data makes it difficult to learn effective translations without additional linguistic insights [61].

4.4.7 Experiments and Results of Parts-Of-Speech Integration

Inspired by the MB feature-integrated NMT models proposed by Supriya et al. [59], the OpenNMT-py toolkit is employed for building POS feature-enhanced systems using the *KT2* and *KanSan* parallel corpora. The hyperparameter settings follow those used in Supriya et al.'s experiments to maintain consistency and comparability. To incorporate POS features into the Kan–Tul and Kan–San NMT systems, the best-performing POS tagger, as detailed in Section 4.4.5, is utilized. This integration aims to improve translation quality by enriching the input with syntactic information derived from accurate POS tagging. Table 31 shows the performance of NMT models before and after incorporating linguistic features.

4.5 Exploration of Orthographies for Enhancing Neural Machine Translation of Under-resourced Languages

Conventionally, NMT models known as bilingual MT models are trained using bilingual parallel corpus and separate models are required for each language pair. This approach requires a substantial amount of storage and computational efforts to develop and maintain multiple bilingual translation models. To overcome these bottlenecks, MT researchers developed MuNMT models that provide a single MT framework for translation between multiple languages [62]. However, the major threat to the performance of MuNMT models is the use of different scripts across the language pairs. Different scripts across multiple languages cause high vocabulary fragmentation, as each script forms a separate word or subword vocabulary, leading to poor cross-lingual embedding sharing [63]. This can be addressed by script normalization - a technique that converts

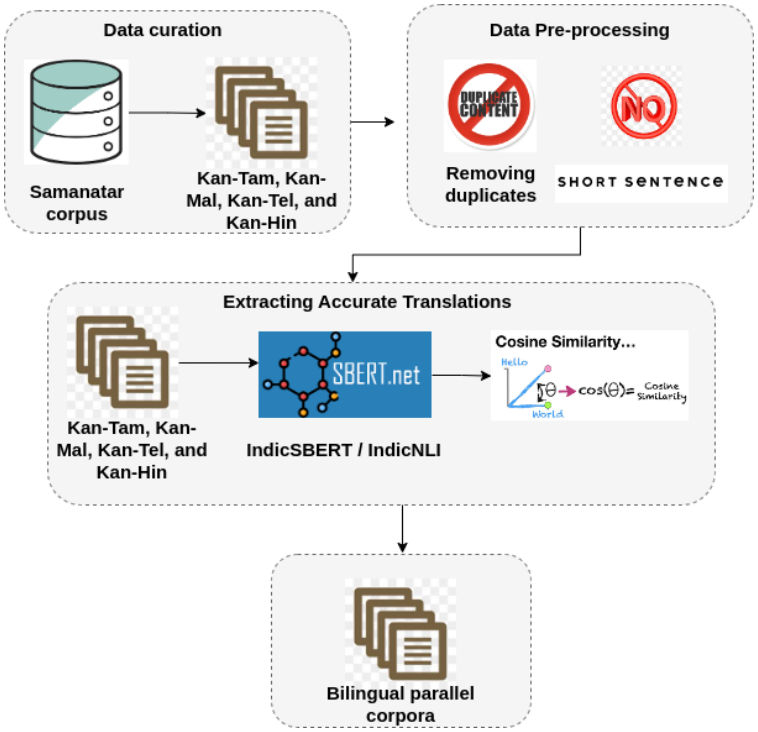


Figure 13: Framework of the bilingual parallel corpora construction

diverse scripts or orthographies of text into a unified script. As closely-related languages share similar phonetic characteristics, mapping phonemes into a unified script offers the advantage of representing similar words at the phoneme level facilitating the identification of linguistic connections aiding cross-lingual information transfer across related languages.

4.5.1 Bilingual Parallel Corpus Construction

Parallel corpus consisting of aligned sentence pairs in two languages which are translations of each other is fundamental to build NMT systems and good quality parallel corpus of large volume plays a crucial role in the performance of NMT models. However, collecting such large volume of parallel corpus for under-resourced languages is challenging due to lack of resources. Moreover, morphological richness of the languages complicates parallel corpus preparation because of their complex nature, making it difficult to match equivalent words accurately across languages. Therefore, as part of this work, bilingual parallel corpora of four languages: Tamil, Telugu, Malayalam, and Hindi, paired with Kannada and multilingual parallel corpus by combining this bilingual corpora and *KanSan* parallel corpus, are constructed and steps involved in the construction are described in the following subsections.

The process of building bilingual parallel corpora involves selecting four language pairs:

Kannada-Tamil (Kan-Tam), Kannada-Malayalam (Kan-Mal), Kannada-Telugu (Kan-Tel), Kannada-Hindi (Kan-Hin), and Kannada-Sanskrit (Kan-San) from Samanantar corpus and Kan-San from *KanSan* parallel corpus. A brief description of these corpora is given below:

- ***Samanantar* -** is the most extensive publicly accessible parallel corpus for eleven Indian languages comprising sentence pairs obtained by web crawling, document-level optical character recognition to extract sentences from scanned documents, and multilingual representation models for sentence alignment. However, it contains numerous duplicate sentences and incorrect translations as sentence pairs are collected from different sources. This can degrade the MT performance and limit the dataset's full potential in training MT systems [64].
- ***KanSan* -** is a first Kan-San bilingual parallel corpus created specifically for MT systems by Hegde et al., [10]. It includes manually aligned document-level data from Mann Ki Baath[36] and monolingual Sanskrit sentences taken from Sanskrit Wikipedia that have been translated into Kannada with the help of linguists. Along with verification and revisions, Google Translate was also utilized for corpus building. The required pre-processing tasks, such as eliminating duplicates and excessive spaces are also employed to clean the data.

For the Kan-San pair, Train and Test sets are taken from the *KanSan* corpus, ensuring that the samples used for training and testing matches with the experiments conducted by Hegde et al., [10], which aids in evaluating translation systems consistently. As the Kan-San sentence pairs have already been pre-processed, the data pre-processing and preparation are performed solely on the sentence pairs extracted from the Samanantar corpus. The steps involved in constructing bilingual parallel corpora are depicted in Figure 13 and are described in the below subsections.

- **Data Pre-processing** - To prevent any performance degradation in MT systems, it is imperative to remove the duplicate sentences in the chosen sentence pairs. Instead of obtaining unique sentence pairs to construct the parallel corpus, unique source and corresponding target sentences are extracted from the source and target texts, respectively.
- **Extracting Accurate Translations** - To address incorrect translation issue in the parallel corpora, a novel yet simple approach is employed to extract sentence pairs that are semantically related. This is achieved by representing the source and target sentences using pre-trained sentence transformer models and calculating cosine similarity between them. IndicNLI and IndicSBERT - the multilingual pre-trained sentence transformer models pre-trained on Kannada, Tamil, Malayalam,

[36]https://www.pmindia.gov.in/

Telugu and Hindi texts are used to represent the sentences. To extract accurate translations that fail to accurately preserve context, sentence pairs having a cosine similarity of more than 0.75 are eventually chosen. This cutoff point is determined via empirical research and this method significantly improves the quality of the parallel corpora.

4.5.2 Test Sets to Assess Bilingual MT Models

To create the Test sets for assessing the bilingual MT models, 5K sentence pairs in the Kan-Tam, Kan-Mal, Kan-Tel, and Kan-Hin languages are randomly selected from the Samanantar corpus. These sentence pairs go through the required pre-processing to ensure the alignment and consistency. This procedure produced 1,000 non-overlapping sentence pairs for every language pair that makes up the Test set.

4.5.3 Construction of Multilingual Parallel Corpus

This work uses, newly constructed bilingual parallel corpora and *KanSan* corpus to build a multilingual parallel corpus, where Kannada sentences are aligned with Tamil, Malayalam, Telugu, Hindi, and Sanskrit sentences. This corpus enables the implementation of one-to-many or many-to-one MuNMT models. Following the methodology inspired by Costa et al., [65], a prefix tagging is employed in this work. This involves adding a special token to both source and target sentences to explicitly denote their respective source and target languages. This method improves the structuring of the multilingual corpus and makes it easier to distinguish between languages while training and assessing translation models.

4.5.4 Methodology for Translation

To train bilingual MT models, this work utilizes newly constructed bilingual parallel corpora and *KanSan* corpus, whereas newly constructed multilingual corpus is used to train MuNMT models. Script normalization is also explored as an additional pre-processing technique and backtransliteration as an additional post-processing technique to enhance the translation performances of the learning models.

SentencePiece Tokenization

SentencePiece - a sub-tokenization technique, provides efficient unsupervised tokenization and detokenization enabling sub-word tokenization for neural text production systems [66]. It supports methods like byte-pair encoding and uni-gram language models,

allowing models to be trained on monolingual corpora without language-specific preprocessing.

Transliteration into Romanized Script

Romanization converts text from various scripts into the Latin alphabet, enabling easier comparison, processing, and translation of languages with similar linguistic features but different writing systems. In this work, romanization is performed using indic-trans[37] library to convert texts from their native scripts to Roman script.

International Phonetic Alphabet Transcription

IPA provides a comprehensive set of symbols to represent the sounds of spoken languages, including vowels, consonants, and prosodic features such as stress and intonation. This work employs the Epitran library[38], a grapheme-to-phoneme transducer supporting 61 languages, including Hindi, Tamil, Telugu, and Malayalam. The library takes words as input and provides phonetic transcription in IPA format. However, it does not support Kannada and Sanskrit. This limitation highlights the need for additional tools to handle Kannada and Sanskrit languages effectively. Therefore, Aksharamukha[39] library is used to convert Kannada and Sanskrit text into IPA.

Sequence-to-Sequence Model with Transformers for MT

The seq2seq paradigm, as explained in Section 4.1.2.4, is used along with transformer architecture which enhances this paradigm with self-attention, allowing the model to efficiently capture dependencies across different positions in the input sequence without relying on recurrence.

Universal Encoder-Decoder for MuNMT

MuNMT models are trained on multilingual parallel corpora, enabling a single model to translate multiple languages by utilizing a shared embedding space that captures common semantics across source text/languages [67]. Instead of having separate encoders and decoders for each language pair, the shared space for the encoder allows the decoder to generate sentences in the desired target language. The model uses special tokens to indicate the source-target language pairs, guiding the translation process during encoding.

[37] https://github.com/libindic/indic-trans
[38] https://pypi.org/project/epitran/
[39] https://pypi.org/project/aksharamukha/

Table 32: Performances of the bilingual MT systems before and after obtaining unique sentence pairs

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Before obtaining unique sentence pairs					
Kan-Tam	10.78	28.70	26.98	20.62	0.86
Kan-Mal	09.98	27.93	25.03	19.12	0.88
Kan-Tel	10.05	28.22	26.81	20.86	0.86
Kan-Hin	**11.68**	21.9	30.07	21.72	0.86
Tam-Kan	10.83	27.61	26.50	20.80	0.86
Mal-Kan	**11.78**	27.35	25.01	21.93	0.86
Tel-Kan	09.20	26.88	24.62	19.55	0.88
Hin-Kan	10.54	25.74	23.70	20.06	0.86
After obtaining unique sentence pairs					
Kan-Tam	12.24	33.36	30.85	28.36	0.81
Kan-Mal	**13.62**	31.26	30.01	21.68	0.86
Kan-Tel	13.34	33.92	31.86	28.02	0.80
Kan-Hin	12.72	32.66	30.24	27.36	0.82
Tam-Kan	13.82	34.32	31.65	27.12	0.80
Mal-Kan	**14.28**	34.11	31.10	28.40	0.80
Tel-Kan	12.65	32.08	31.11	26.94	0.80
Hin-Kan	11.91	30.68	29.63	20.03	0.86

4.5.5 Experiments and Results

Several experiments are conducted by implementing bilingual MT and MuNMT models with training data in different orthographies. To evaluate the performances of BMT and MuNMT models using multiple evaluation metrics are employed. BLEU scores assess translation quality by comparing word n-grams between candidate and reference texts, while CHRF measures precision, recall, and F-scores at the character level. CHRF++ extends CHRF by integrating word n-grams and Metric for Evaluation of Translation with Explicit Ordering (METEOR) evaluates translations based on unigram matches. Additionally, Word Error Rate (WER) quantifies the percentage of words that need to be modified to align the candidate translation with the reference. These experiments utilize OpenNMT-py and OpenNMT-py parameters are used based on the paper Hegde et al., [10] and the OpenNMT-py is configured using these parameters exhibited the best performance in their work for Kan-San and San-Kan MT systems.

Performances of the BMT models in terms of BLEU scores are shown in Table 32 before and after extracting unique sentence pairs. These results show that translation performance, as measured by BLEU scores, improved significantly after obtaining unique sentence pairs from the Samanantar corpus. This process has reduced biases and enhanced translation quality. Table 33 shows the performances of the BMT models after excluding incorrect translations in their native, romanized, and IPA scripts. It may be noted that, during demonstrating the performances of BMT models after removing incorrect translations include Kan-San language pair as well along with four

Table 33: Performances of the bilingual MT systems excluding incorrect translations

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Native script					
Kan-Tam	13.60	36.14	34.21	29.38	0.78
Kan-Mal	12.11	32.63	30.88	26.08	0.80
Kan-Tel	14.36	36.02	33.42	29.10	0.79
Kan-Hin	13.61	33.68	32.63	27.36	0.80
Kan-San	**15.40**	32.09	30.54	25.22	0.81
Tam-Kan	15.81	38.65	35.84	30.61	0.78
Mal-Kan	13.16	35.22	33.64	27.8	0.81
Tel-Kan	11.63	33.28	30.81	26.72	0.80
Hin-Kan	12.04	32.61	30.93	26.24	0.80
San-Kan	**17.97**	34.13	32.06	28.30	0.80
Romanized script					
Kan-Tam	**18.60**	40.37	39.66	30.25	0.76
Kan-Mal	14.96	37.61	36.83	29.96	0.80
Kan-Tel	14.98	36.29	33.50	29.61	0.80
Kan-Hin	13.82	36.83	33.69	30.86	0.80
Kan-San	17.59	40.72	38.68	29.90	0.76
Tam-Kan	**19.92**	42.72	40.98	31.17	0.76
Mal-Kan	15.61	48.85	35.64	30.60	0.80
Tel-Kan	15.81	33.63	30.92	26.83	0.80
Hin-Kan	12.23	33.76	31.02	27.31	0.79
San-Kan	19.84	40.77	42.29	32.30	0.77
IPA transcription					
Kan-Tam	14.80	27.91	27.86	22.92	0.80
Kan-Mal	11.16	26.71	25.71	20.63	0.86
Kan-Tel	**16.83**	30.68	29.69	28.24	0.80
Kan-Hin	14.94	28.07	27.87	26.70	0.80
Kan-San	14.04	32.35	30.81	25.60	0.78
Tam-Kan	**16.62**	42.58	40.90	29.70	0.78
Mal-Kan	12.35	27.63	26.07	19.63	0.76
Tel-Kan	15.78	28.70	28.00	26.81	0.80
Hin-Kan	13.82	28.03	26.98	20.96	0.80
San-Kan	16.03	31.96	30.84	26.80	0.78

Table 34: Performances of the best performing BMT systems after backtransliteration of the generated text from romanized and IPA scripts

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Backtransliterated from romanized script					
Kan-Tam	17.32	41.75	40.66	29.33	0.73
Kan-Mal	14.03	36.32	35.46	28.53	0.76
Kan-Tel	15.33	37.55	35.47	28.74	0.76
Kan-Hin	15.25	37.91	36.18	28.77	0.72
Kan-San	**18.17**	41.31	40.39	32.85	0.72
Tam-Kan	**18.79**	40.67	38.76	31.61	0.77
Mal-Kan	16.75	38.84	36.19	38.53	0.73
Tel-Kan	14.74	36.29	34.36	27.56	0.78
Hin-Kan	13.24	37.39	35.43	27.97	0.78
San-Kan	18.28	41.61	40.76	33.67	0.71
Backtransliterated from IPA transcript					
Kan-Tam	13.35	28.21	27.39	21.44	0.78
Kan-Mal	10.75	26.35	25.12	20.62	0.80
Kan-Tel	17.00	30.42	29.56	29.99	0.77
Kan-Hin	14.71	27.37	25.20	27.94	0.78
Kan-San	**18.66**	31.76	29.20	24.45	0.77
Tam-Kan	15.34	41.33	39.85	28.08	0.78
Mal-Kan	13.16	28.18	26.52	18.27	0.78
Tel-Kan	16.15	27.67	26.76	25.01	0.77
Hin-Kan	14.28	28.97	26.43	21.39	0.78
San-Kan	**19.24**	30.38	29.15	25.84	0.77

Table 35: Performances of MuNMT systems before removing duplicates

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Kan-Tam	14.81	36.02	34.81	28.57	0.78
Kan-Mal	12.15	35.57	33.65	26.37	0.80
Kan-Tel	16.24	36.93	33.01	26.62	0.80
Kan-Hin	**18.19**	37.61	34.89	28.83	0.80
Kan-San	16.04	35.08	33.51	26.05	0.79
Tam-Kan	15.70	35.31	32.24	25.34	0.78
Mal-Kan	13.65	35.98	32.33	25.50	0.80
Tel-Kan	15.83	36.88	33.70	26.58	0.78
Hin-Kan	16.35	36.96	33.68	26.32	0.77
San-Kan	17.92	36.84	34.45	26.32	0.77

Table 36: Performances of the MuNMT systems with native scripts, romanized versions, and IPA transcriptions after removing duplicates

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Native script					
Kan-Tam	16.32	38.68	36.95	30.28	0.73
Kan-Mal	13.90	38.79	36.18	28.63	0.73
Kan-Tel	18.61	39.96	36.74	32.86	0.73
Kan-Hin	19.65	40.83	38.24	32.90	0.72
Kan-San	**20.93**	43.20	40.98	34.79	0.71
Tam-Kan	20.80	42.75	40.87	34.92	0.71
Mal-Kan	15.07	39.82	38.64	32.34	0.71
Tel-Kan	17.81	33.91	30.38	28.03	0.73
Hin-Kan	16.64	33.74	30.46	27.34	0.73
San-Kan	**22.68**	44.06	40.97	35.80	0.71
Romanized script					
Kan-Tam	17.01	39.61	36.32	32.59	0.71
Kan-Mal	13.86	36.86	34.20	32.64	0.72
Kan-Tel	19.96	32.24	30.63	30.25	0.70
Kan-Hin	20.65	40.92	38.79	36.86	0.70
Kan-San	**21.20**	43.65	42.44	35.93	0.68
Tam-Kan	22.69	42.60	40.98	36.09	0.68
Mal-Kan	18.36	36.08	33.51	34.54	0.71
Tel-Kan	21.82	30.50	28.80	27.01	0.68
Hin-Kan	18.44	31.89	29.96	29.73	0.71
San-Kan	**23.50**	43.58	41.90	36.35	0.67
IPA transcript					
Kan-Tam	14.96.	38.51	36.21	29.65	0.80
Kan-Mal	12.63	33.84	31.64	29.33	0.81
Kan-Tel	18.29	33.21	30.29	28.92	0.78
Kan-Hin	16.80	38.65	36.73	30.71	0.81
Kan-San	**20.46**	41.16	39.54	30.45	0.76
Tam-Kan	**23.75**	40.96	38.80	27.34	0.76
Mal-Kan	12.68	36.82	32.01	34.69	0.81
Tel-Kan	18.06	31.93	28.38	25.29	0.77
Hin-Kan	15.92	32.07	30.96	30.50	0.80
San-Kan	21.85	40.63	39.03	32.86	0.75

Table 37: Performances of the MuNMT systems with native scripts, romanized versions, and IPA transcriptions after excluding incorrect translations

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Native script					
Kan-Tam	20.27	38.32	36.25	30.18	0.70
Kan-Mal	18.09	36.75	33.63	37.67	0.71
Kan-Tel	22.78	41.67	39.17	38.51	0.69
Kan-Hin	25.60	45.03	42.79	39.75	0.68
Kan-San	**26.01**	46.22	44.15	39.44	0.68
Tam-Kan	25.02	44.15	41.18	33.17	0.68
Mal-Kan	21.37	40.11	38.91	30.24	0.70
Tel-Kan	20.50	40.44	39.31	32.12	0.70
Hin-Kan	23.29	42.53	39.69	36.40	0.70
San-Kan	**27.23**	44.79	41.67	39.23	0.68
Romanized script					
Kan-Tam	21.30	41.65	40.75	32.43	0.69
Kan-Mal	20.24	40.27	38.47	30.3	0.70
Kan-Tel	25.56	46.87	44.29	37.11	0.68
Kan-Hin	26.70	48.50	46.11	36.53	0.68
Kan-San	**30.00**	52.95	50.06	42.99	0.65
Tam-Kan	27.93	50.65	49.76	38.56	0.67
Mal-Kan	22.86	41.61	40.20	36.42	0.68
Tel-Kan	21.01	40.47	38.63	36.00	0.69
Hin-Kan	24.92	45.75	42.36	36.62	0.68
San-Kan	**32.20**	56.5	53.9	43.12	0.60
IPA script					
Kan-Tam	20.85	38.69	36.91	31.74	0.68
Kan-Mal	19.35	35.18	32.18	30.28	0.68
Kan-Tel	23.19	41.17	40.77	39.28	0.67
Kan-Hin	24.84	42.31	40.85	32.08	0.66
Kan-San	**27.15**	50.39	48.08	43.16	0.60
Tam-Kan	23.79	46.2	44.03	37.31	0.67
Mal-Kan	20.67	38.16	36.52	38.82	0.68
Tel-Kan	22.25	40.9	38.11	38.4	0.68
Hin-Kan	21.76	37.63	35.61	38.98	0.68
San-Kan	**27.58**	51.25	50.53	42.01	0.70

Table 38: Performances of the best performing MuNMT systems after backtransliteration of romanized and IPA versions of the text

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Backtransliterated from romanized script					
Kan-Tam	21.32	42.65	40.78	31.94	0.68
Kan-Mal	18.79	41.21	40.60	30.89	0.69
Kan-Tel	24.70	46.24	43.29	37.29	0.66
Kan-Hin	27.93	48.36	44.20	36.34	0.60
Kan-San	**31.37**	51.22	50.65	41.86	0.60
Tam-Kan	25.15	50.88	48.52	36.39	0.66
Mal-Kan	21.15	41.20	39.65	35.46	0.68
Tel-Kan	20.54	39.93	38.02	34.54	0.68
Hin-Kan	24.20	44.27	42.01	34.43	0.67
San-Kan	**31.68**	56.09	54.86	42.44	0.60
Backtransliterated from IPA transcript					
Kan-Tam	19.11	39.99	38.75	30.45	0.68
Kan-Mal	18.53	36.67	34.66	29.39	0.68
Kan-Tel	24.99	43.75	42.83	37.93	0.67
Kan-Hin	23.56	44.53	42.67	33.47	0.67
Kan-San	**27.12**	51.55	50.78	42.72	0.66
Tam-Kan	20.13	47.72	46.44	35.59	0.68
Mal-Kan	19.45	40.47	38.38	36.21	0.68
Tel-Kan	22.34	39.49	37.35	36.22	0.67
Hin-Kan	21.20	40.17	38.38	37.74	0.68
San-Kan	**26.86**	51.22	50.23	42.83	0.66

language pairs in bilingual parallel corpora. Further, Table 34 shows the performances of the BMT models after backtransliteration of the generated text from romanized and IPA versions. From the table, it is clear that for few language pairs the backtransliteration has performed well leaving few language pairs on the lower side, compared to the performances reported in Table 33.

The approach proposed by Ha et al., [67] integrates a multilingual setting into a unified encoder-decoder framework, allowing seamless extension of the BMT model to MuNMT without altering hyperparameters. This unified method simplifies training by eliminating the need for special adjustments and leverages the existing OpenNMT framework used for BMT. As a result, it provides a streamlined MT approach to handle multiple languages within the same model. Table 35 reports the performances of MuNMT systems before obtaining unique sentence pairs in their native scripts. Further, it also illustrates that the BLEU scores are lower due to the presence of duplicates and incorrect translations. Additionally, for some language pairs, the BLEU scores are even lower than those achieved by BMT systems. Table 36 presents the performances in terms of BLEU scores of the MuNMT systems trained using text in its native, romanized, and IPA versions after obtaining unique sentence pairs. From the table, it is clear that after obtaining unique sentence pairs, the BLEU scores are significantly increased indicating the impact of duplicate removal on translation performance.

Performances of the MuNMT models trained with native, romanized, and IPA ver-

sions of the text after removing incorrect translations are reported in Table 37. This table demonstrates that removing incorrect translations significantly improves the performance of MuNMT systems, leading to more accurate translations. To ensure the consistency in evaluation and to conduct experiments that are more accurately reflect real-world scenarios, translations generated by the best-performing models (MuNMT models trained after excluding incorrect translations) are backtransliterated into their respective native scripts. This approach aims to maintain the integrity and realism of the evaluation process by aligning the translations with their native scripts. Table 38 shows the performances obtained after backtransliteration of generated translations obtained from the best-performing MuNMT models.

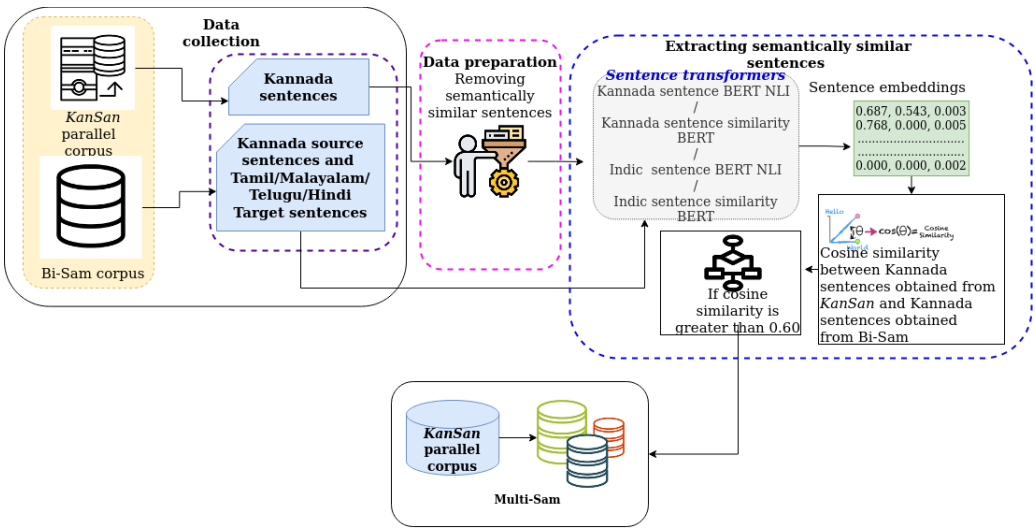

Data collection
KanSan parallel corpus
Bi-Sam corpus
Kannada sentences
Kannada source sentences and Tamil/Malayalam/ Telugu/Hindi Target sentences
Data preparation
Removing semantically similar sentences
Extracting semantically similar sentences
Sentence transformers
Kannada sentence BERT NLI / Kannada sentence similarity BERT / Indic sentence BERT NLI / Indic sentence similarity BERT
Sentence embeddings
0.687, 0.543, 0.003
0.768, 0.000, 0.005
0.000, 0.000, 0.002
Cosine similarity between Kannada sentences obtained from KanSan and Kannada sentences obtained from Bi-Sam
If cosine similarity is greater than 0.60
KanSan parallel corpus
Multi-Sam

Figure 14: Framework of the proposed methodology for constructing multilingual corpus of semantically similar sentences

4.6 Bridging Under-resourced Languages with Semantic Multilingual Neural Translation

To build MuNMT systems, a multilingual corpus with sufficiently large amount of training data is crucial. Although there are many strategies to construct multilingual corpora, most of them rely heavily on the size of the training data, highlighting the need for further research in this area. To address the challenges of building MuNMT systems, in this work a simple yet effective approach to build bilingual corpora and using it to build multilingual corpus thereby implementing BMT and MuNMT systems to showcase the enhanced translation performance of the MT models. This study utilizes bilingual parallel corpora developed in our previous work, with the Test sets

Table 39: Number of words and unique words curated for IST+IndicSBERT

Language pair	Language	# of words	# of unique words
Kan-Tam	Kannada	941,203	181,407
	Tamil	967,042	187,835
Kan-Mal	Kannada	1,080,615	205,217
	Malayalam	1,027,831	245,161
Kan-Tel	Kannada	1,049,507	204,137
	Telugu	1,071,750	199,509
Kan-Hin	Kannada	1,204,775	223,774
	Hindi	1,881,036	88,639
Kan-San	Kannada	593,601	119,519
	Sanskrit	717,840	696,155

for evaluating the BMT models also sourced from the same, as detailed in Sections 4.5.1 and 4.5.2.

4.6.1 Construction of Multilingual Parallel Corpus

The proposed multilingual parallel corpus - *Multi-Sam*, is constructed by considering relatedness of the languages and languages that share semantic similarities between source-target sentence pairs. Leveraging these concepts ensure that sentence pairs in the resulting *Multi-Sam* corpus belongs to same/similar domain. Earlier MuNMT research indicates that systems trained using multi-parallel corpus containing the same source sentences and their translations across multiple languages perform better even with limited data. However, constructing such corpora for under-resourced languages is challenging as it demands tools required to extract sentence pairs across different languages and linguists for verification and correction. To address this challenge, the *Multi-Sam* corpus is constructed. This corpus is built by considering the semantic similarity between source sentences in Kannada (common source language) and the corresponding target sentences across multiple languages. This approach aims to enhance translation quality even in resource-constrained settings.

The framework of the proposed methodology to build *Multi-Sam* corpus is visualized in Figure. 14. As illustrated in the figure, data preparation module helps in preparing the by removing similar sentences from *KanSan* corpus, while extracting semantically similar sentences module makes use of IndicSBERT pre-trained model and helps in extracting semantically similar sentence pairs from *Bi-Sam* corpus. Descriptions of each of these modules are given below:

– **Data Preparation** - involves considering 1,00,171 Kannada sentences from the *KanSan* parallel corpus and all the sentence pairs belonging to four language pairs from the *Bi-Sam* corpus. To exclude semantically similar Kannada sentences, sentence embeddings are computed for Kannada sentences using IndicSBERT

Table 40: Statistics of *Multi-Sam*

Type of data	# of sentence pairs	# of tokens in multilingual text	# of tokens in Kannada text	# of unique words in multilingual text	# of unique words in Kannada text
Before obtaining unique sentence pairs	1,300,171	10,929,934	8,594,730	1,065,910	731,855
After obtaining unique sentence pairs	942,976	8,902,347	7,836,572	1,001,170	876,740
After excluding wrong translations	8,43,109	7,091,432	6,083,168	994,934	729,648
With semantically similar sentence pairs	8,16,953	6,463,147	5,695,840	973,801	541,604

pre-trained model and similarity of these embeddings is computed using cosine similarity measures. If the cosine similarity value of the sentences is greater than or equal to 0.85, then only one sentence is retained; otherwise, both the sentences are retained, to exclude semantically similar sentences. The threshold for similarity score is decided based on extensive empirical studies to ensure the exclusion of same/similar Kannada sentences.

– **Extracting semantically similar sentence pairs** - involves computing sentence embeddings for Kannada sentences retained in data preparation step and Kannada sentences in bilingual parallel corpora, using pre-trained sentence transformer models: IndicNLI, IndicSBERT, Kannada sentence BERT NLI (KannadaNLI)[40], and Kannada Sentence Similarity BERT (KannadaSBERT)[41].

In order to select Kannada sentences from bilingual parallel corpora which are semantically similar to Kannada sentences in *KanSan* corpus, cosine similarity between the sentence embeddings of Kannada sentences in *KanSan* corpus and Kannada sentences in newly constructed bilingual parallel corpora. are computed. Kannada sentences and their corresponding target sentences in *Bi-Sam* corpus having similarity score greater than 0.60 makes entry to *Multi-Sam* corpus. Further, Kan-San sentence pairs are also added to this corpus. Number of sentence pairs from IndicSBERT are more compared to other sentence transformers. Therefore, only the sentence pairs obtained from IndicSBERT for IST (sentences extracted using IndicBERT model) (IST+IndicSBERT) are used for further experiments and Table 39 shows the total number of words and unique words for IST+IndicSBERT.

The above process has generated Multi-Sam - a multilingual parallel corpus of Kannada

[40]https://huggingface.co/l3cube-pune/kannada-sentence-bert-nli
[41]https://huggingface.co/l3cube-pune/kannada-sentence-similarity-sbert

Table 41: Performances of the BNMT systems before and after obtaining unique sentence pairs and excluding wrong translations for Kan-Tam, Kan-Mal, Kan-Tel, Kan-Hin language pairs and vice versa

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Before obtaining unique sentence pairs					
Kan-Tam	10.78	28.70	26.98	20.62	0.86
Kan-Mal	09.98	27.93	25.03	19.12	0.88
Kan-Tel	10.05	28.22	26.81	20.86	0.86
Kan-Hin	**11.68**	21.90	30.07	21.72	0.86
Tam-Kan	10.83	27.61	26.50	20.80	0.86
Mal-Kan	**11.78**	27.35	25.01	21.93	0.86
Tel-Kan	09.20	26.88	24.62	19.55	0.88
Hin-Kan	10.54	25.74	23.70	20.06	0.86
After obtaining unique sentence pairs					
Kan-Tam	12.24	33.36	30.85	28.36	0.81
Kan-Mal	11.62	31.26	30.01	21.68	0.86
Kan-Tel	**13.34**	33.92	31.86	28.02	0.80
Kan-Hin	12.72	32.66	30.24	27.36	0.82
Tam-Kan	**13.82**	34.32	31.65	27.12	0.80
Mal-Kan	12.28	34.11	31.1	28.4	0.80
Tel-Kan	12.65	32.08	31.11	26.94	0.80
Hin-Kan	10.01	30.68	29.63	20.03	0.86
After excluding wrong translations					
Kan-Tam	13.60	36.14	34.21	29.38	0.78
Kan-Mal	12.11	32.63	30.88	26.08	0.80
Kan-Tel	**14.36**	36.02	33.42	29.10	0.79
Kan-Hin	13.61	33.68	32.63	27.36	0.80
Tam-Kan	**15.81**	38.65	35.84	30.61	0.78
Mal-Kan	13.16	35.22	33.64	27.80	0.81
Tel-Kan	11.63	33.28	30.81	26.72	0.80
Hin-Kan	12.04	32.61	30.93	26.24	0.80

sentences paired with Tamil, Malayalam, Telugu, Hindi, and Sanskrit sentences and the statistics of the *Multi-Sam* corpus are shown in Table 40.

4.6.2 Methodology for Translation

MT framework and the hyperparameters and their values for both bilingual MT and MuNMT models are employed as described in Section 4.5.4.

Table 42: Performances of the BMT systems for *KanSan* parallel corpus

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Kan-San (Baseline)	09.84	26.37	25.66	18.25	0.88
Kan-San (Proposed)	**15.40**	32.09	30.54	25.22	0.81
San-Kan (Baseline)	12.63	30.80	28.16	26.58	0.80
San-Kan (Proposed)	**17.97**	34.13	32.06	28.30	0.80

Table 43: Performances of the MuNMT systems before and after obtaining unique sentence pairs for *Multi-Sam* corpus

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
Before obtaining unique sentence pairs					
Kan-Tam	13.60	36.14	34.21	29.38	0.78
Kan-Mal	12.11	32.63	30.88	26.08	0.80
Kan-Tel	14.36	36.02	33.42	29.10	0.79
Kan-Hin	13.61	33.68	32.63	27.36	0.80
Kan-San	**16.04**	35.08	33.51	26.05	0.79
Tam-Kan	15.81	38.65	35.84	30.61	0.78
Mal-Kan	13.16	35.22	33.64	27.80	0.81
Tel-Kan	11.63	33.28	30.81	26.72	0.80
Hin-Kan	12.04	32.61	30.93	26.24	0.80
San-Kan	**17.92**	36.84	34.45	26.32	0.78
After obtaining unique sentence pairs					
Kan-Tam	14.81	36.02	34.81	28.57	0.78
Kan-Mal	12.15	35.57	33.65	26.37	0.80
Kan-Tel	16.24	36.93	33.01	26.62	0.80
Kan-Hin	18.19	37.61	34.89	28.83	0.80
Kan-San	**20.93**	43.20	40.98	34.79	0.71
Tam-Kan	15.70	35.31	32.24	25.34	0.78
Mal-Kan	13.65	35.98	32.33	25.50	0.80
Tel-Kan	15.83	36.88	33.70	26.58	0.78
Hin-Kan	16.35	36.96	33.68	26.32	0.77
San-Kan	**22.68**	44.06	40.97	35.80	0.71

Table 44: Performances of the MuNMT systems after excluding wrong translations and extracting semantically similar sentence pairs for Mutli-Sam corpus

Language Pair	BLEU	CHRF	CHRF++	METEOR	WER
After excluding wrong translations					
Kan-Tam	16.32	38.68	36.95	30.28	0.73
Kan-Mal	13.90	38.79	36.18	28.63	0.73
Kan-Tel	18.61	39.96	36.74	32.86	0.73
Kan-Hin	19.65	40.83	38.24	32.90	0.72
Kan-San	**26.01**	46.22	44.15	39.44	0.68
Tam-Kan	20.80	42.75	40.87	34.92	0.71
Mal-Kan	15.07	39.82	38.64	32.34	0.71
Tel-Kan	17.81	33.91	30.38	28.03	0.73
Hin-Kan	16.64	33.74	30.46	27.34	0.73
San-Kan	**27.23**	44.79	41.67	39.23	0.68
After extracting semantically similar sentences pairs					
Kan-Tam	20.49	39.36	37.27	31.36	0.68
Kan-Mal	19.20	38.31	36.33	29.06	0.70
Kan-Tel	23.60	41.33	39.14	33.92	0.67
Kan-Hin	25.54	43.04	40.25	34.17	0.65
Kan-San	**28.12**	48.16	46.38	38.17	0.62
Tam-Kan	26.68	46.35	44.94	35.98	0.65
Mal-Kan	23.02	40.56	38.22	33.12	0.67
Tel-Kan	22.17	40.39	38.72	32.28	0.67
Hin-Kan	24.04	42.59	40.28	33.34	0.66
San-Kan	**31.12**	48.62	46.65	40.62	0.60

4.6.3 Experiments and Results

Evaluation metrics to evaluate bilingual MT and MuNMT models are used same as described in Section 4.6.3. The translation performances of BNMT systems before and after obtaining unique sentence pairs and also after excluding wrong translations from the bilingual parallel corpora are shown in Table 41. As shown in the table, obtaining unique sentence pairs from each language pair in the bilingual parallel corpora improves the BLEU score and similar trends are observed in other evaluation metrics as well. Table 43 illustrates the performances of MuNMT models before and after obtaining unique sentence pairs for *Multi-Sam* and Table 44 shows the MuNMT performances after using the newly created *Multi-Sam* corpus with semantically similar sentence pairs.

4.7 Automatic Post Editing for Neural Machine Translation

Transformer-based NMT models with an encoder-decoder architecture have become the state-of-the-art for efficient translation, particularly for morphologically less complex non-agglutinative languages with high-resources [68]. However, there are several obstacles to overcome when developing high-performance NMT systems for morphologically rich agglutinative under-resourced languages. This can be addressed by incorporating additional techniques that can be beneficial specifically building customized NMT models for morphologically rich agglutinative under-resourced languages.

Automatic Post Editing (APE) in NLP refers to the process of using ML algorithms or rule-based systems to automatically correct errors or improve the quality of machine-generated text output. It is an area of research aiming at exploring methods for learning from human post-edited data and applying the results to produce better MT output and can also be used as an additional post-processing technique to improve the translation performance of the MT models [31]. In this study, APE is incorporated to further refine and correct translations. By leveraging APE, the models aim to provide more accurate and contextually appropriate translations. This customization helps to address the specific challenges posed by the unique structures of the Kannada, Tulu, and Sanskrit languages.

4.7.1 Proposed Method

The nature of APE necessitates a triplet format and hence, encourages to use dual-encoders while implementing NMT systems rather than using single encoder. The proposed model typically comprises two encoder components, each leveraging transformer-based neural networks to process and transform input sequences into high-dimensional vectors. These vectors are then compared, often via a dot product or a learned metric, to determine the inherent similarity or relationship between the two inputs. This approach proves particularly effective in tasks demanding an understanding of the inter-connectivity between two pieces of text or

between text and image modalities, capitalizing on the transformer's strength in capturing contextual information within each sequence.

In this work, three different types of APE architectures are explored and description of the architectures is given below:

- **Concatenation** - is a method that combines the representations from two encoders by joining them along the feature dimension, enabling the decoder to access the full information from both the source and the machine-translated text. This approach is simple and intuitive, offering a direct way to merge the data. However, it may not fully capture the nuanced relationships or inter-dependencies between the source and target texts. As a result, while concatenation allows for a complete view of the information, it may miss subtle connections that could improve translation quality. Therefore, while effective in some cases, this method may have limitations when it comes to maintaining the contextual coherence between both texts [69].

- **Cross-attention** - is a technique where the decoder attends to the representations from both the source and machine-translated texts separately, enabling it to dynamically integrate information from each encoder during the decoding process. This method allows the model to focus on the most relevant parts of both texts at each step of generation, improving the quality of the output. By doing so, cross-attention can lead to more nuanced and contextually appropriate edits, as it better captures the relationships between the source and target language. Unlike concatenation, which treats the information as a whole, cross-attention helps the model make more informed decisions by considering both sources independently but in parallel. This makes it a powerful tool for producing translations that are more accurate and contextually coherent [70].

- **Feature fusion** - is the proposed method that combines encoder representations using both concatenation and cross-attention techniques. This hybrid architecture allows for a more comprehensive integration of information by merging the strengths of both approaches. The concatenation method joins the representations along the feature dimension, while the cross-attention mechanism enables the decoder to focus on relevant parts of both the source and machine-translated texts independently. By fusing these two representations, the model captures more nuanced and relevant information from the input. As a result, the feature fusion method enhances the overall performance of translation.

4.7.2 Experiments and Results

The experiment was conducted with *KT2* and *KanSan* bilingual corpora by implementing dual-encoder NMT model with a self-attention. The Hyperparameters and their values of

Table 45: Hyperparameters and their values of the proposed dual-encoder model

Parameter	Value
Architecture	DualEncoderTransformer
Dropout	0.3
BatchSize	64
Encoder1EmbedDimensions	100
Encoder1Layers	2
Encoder1AttentionHeads	8
Encoder2EmbedDimensions	100
Encoder2Layers	2
Encoder2AttentionHeads	8
DecoderEmbedDimensions	100
DecoderLayers	2
DecoderAttention Heads	8
NumberofMaxEpochs	10
Optimizer	Adam
LearningRate	0.0005

Table 46: Performance of the proposed dual-encoder decoder model

Language pair	Before APE		APE (with concatenation)		APE (with cross-attention)		APE (with fusion)	
	BLEU	CHRF	BLEU	CHRF	BLEU	CHRF	BLEU	CHRF
Kan-San	09.84	26.37	07.85	24.02	12.05	35.06	12.25	36.00
San-Kan	12.63	30.80	10.30	28.50	15.05	38.82	14.80	39.18
Kan-Tul	24.10	50.29	23.84	47.98	27.90	44.91	28.30	45.04
Tul-Kan	34.15	59.80	34.50	56.03	35.88	47.48	36.58	48.64

the proposed dual-encoder model is shown in Table 45. It may be noted that, to train the proposed dual-encoder decoder model, source text, model generated text (obtained from the best performing model), and target text belonging to Kan-San and Kan-Tul language pairs are considered. The performance of the proposed dual-encoder decoder model is shown in Table 46.

4.8 Performance Comparison

MT systems for morphologically rich under-resourced languages have been gaining increasing attention from the MT research community. These languages present unique challenges due to their complex morphology and limited available data. Encoder-decoder based NMT models, which enable seq2seq processing of text, have become a prominent trend in addressing these challenges. By leveraging DL techniques, these models improve translation quality offering a promising solution for such languages. As a case study, this work focuses on the Kan-San and Kan-Tul language pairs, where MT models are implemented from scratch by constructing parallel corpora for these languages. At the time this research was initiated, no other studies had specifically addressed the development of MT systems for these language pairs. This gap in the research strongly motivated us to explore these languages and contribute to advancing MT solutions for them. The lack of prior work in this area provided a

Table 47: Comparison of performance of MT systems for Kan-San language pair for *KanSan* parallel corpus with augmented text

Models	**Kan-San**		**San-Kan**	
	BLEU	CHRF	BLEU	CHRF
Transformer+BPE	09.84	26.37	12.63	30.80
Google translator	07.13	20.91	12.28	29.00
Model inspired by Supriya et al.,	16.13	30.50	13.04	34.71
Proposed POS integrated model	16.28	31.07	13.24	35.11
APE (with concatenation)	07.85	24.02	10.30	28.50
APE (with cross-attention)	12.05	35.06	15.05	38.82
Proposed APE model	12.25	36.00	14.80	39.18
Proposed MuNMT model (with romanization)	**30.00**	52.95	**32.20**	56.50
Proposed MuNMT model (with the proposed multilingual corpus)	28.12	48.16	31.12	48.62

unique opportunity to develop tailored approaches for these under-resourced languages. Due to lack of publicly available parallel corpora for this language pair, countable works (from the present work) are considered for the performance comparison of Kan-San and Kan-Tul MT systems. The description of these are given in the following subsections.

4.8.1 Kannada-Sanskrit Machine Translation Systems

Performance of the MT baselines for Kan-San language pairs (as mentioned in Table 16), where existing models are implemented and BLEU scores are compared with the performances of Google Translator (as per May 2023). Further, the BLEU scores obtained from top performing baseline are compared with enhanced MT performances in the proposed work. Table 47 gives performance comparison for Kan-San and San-Kan language pairs explored in this work.

4.8.2 Kannada-Tulu Machine Translation Systems

The development of Kan-Tul MT models began with the creation of two versions of parallel corpora. The first version, the *KT1* parallel corpus, while the second version, *KT2*, was constructed to address biases in the initial corpus. The performance of the Kan-Tul MT models was then compared with that of Google Translator (as of June 2024) and the improved Kan-Tul models. This comparison highlights performance differences and is presented in Table 48. These results showcase the advancements made in the Kan-Tul MT models.

Table 48: Comparison of performance of MT systems for Kan-Tul language pair for *KT2* parallel corpus with augmented text

Models	**Kan-Tul**		**Tul-Kan**	
	BLEU	CHRF	BLEU	CHRF
Transformer+BPE	24.10	50.29	34.15	59.80
Google translator	27.35	54.04	35.71	59.90
Model inspired by Supriya et al., with POS integration	30.32	54.69	39.66	62.08
Proposed POS integrated model	**30.35**	55.00	**39.84**	63.58
Model inspired by Supriya et al., with MB integration	24.35	51.96	30.02	49.33
Proposed MB integrated model	24.57	52.03	30.82	49.50
APE (with concatenation)	23.84	47.98	34.50	56.03
APE (with cross-attention)	27.90	44.91	35.88	47.48
Proposed APE model	28.30	45.04	36.58	48.64

5 Conclusion and Future Works

In this work, we propose efficient NMT models for Kan-San and Kan-Tul language pairs, focusing on improving translation performance for these low-resource languages. As part of our objectives, we have manually constructed the *KT1*, *KT2*, and *KanSan* parallel corpora, as well as MB and POS tagging datasets to aid in model training. Additionally, we explore the use of TA techniques to further enhance the performance of NMT models trained on the *KT1*, *KT2*, and *KanSan* parallel corpora. These MB and POS tagging datasets are benchmarked by implementing various learning models to evaluate their effectiveness. We also investigate the impact of different orthographies on MT by implementing both bilingual MT and MuNMT models. A key aspect of this work is the exploration of a simple yet novel multilingual parallel corpus construction strategy, which is empirically tested to assess its impact on translation quality. Furthermore, the integration of MB and POS features during the training of NMT models is examined, showing that these linguistic features significantly enhance the performance of MT systems. Finally, we investigate the use of APE as an additional post-processing technique to further refine the outputs of our NMT models. The empirical studies conducted in this work demonstrate the effectiveness of these strategies in improving translation quality for the Kan-San and Kan-Tul language pairs, contributing to the advancement of machine translation for under-resourced languages.

Looking ahead, future work can focus on expanding the proposed models to include more low-resource language pairs, particularly those that share similar linguistic features or orthographies. Another avenue for future research is the development of more sophisticated bilingual and multilingual corpora, leveraging unsupervised learning techniques and crowd-sourcing for data collection. Additionally, further exploration of advanced linguistic features, such as syntactic trees or semantic information, could further refine translation models. Investigating the potential of pre-trained language models in low-resource settings and combining them with fine-tuning strategies for specific language pairs may also open new opportunities for improving translation performance. Lastly, the integration of real-time

adaptive learning systems and more advanced post-processing techniques, such as neural-based APE models, could lead to even greater improvements in the accuracy and fluency of the translations for these under-resourced languages.

6 Publications

6.1 Publications in Book Chapter

- Hegde, Asha, and Hosahalli Lakshmaiah Shashirekha. "Syllable-Level Morphological Segmentation of Kannada and Tulu Words." Automatic Speech Recognition and Translation for Low Resource Languages (2024): 113-133.

6.2 Publications in International Conferences

- Hegde, Asha, et al. "A study of machine translation models for Kannada-Tulu." Congress on Intelligent Systems. Singapore: Springer Nature Singapore, 2022.
- Hegde, Asha, and Hosahalli Lakshmaiah Shashirekha. "KanSan: Kannada-Sanskrit Parallel Corpus Construction for Machine Translation." International Conference on Speech and Language Technologies for Low-resource Languages. Cham: Springer International Publishing, 2022.
- Asha, Hegde, and Shashirekha Hosahalli Lakshmaiah. "KT2: Kannada-Tulu Parallel Corpus Construction for Neural Machine Translation." Proceedings of the 20th International Conference on Natural Language Processing (ICON). 2023.

6.3 Papers Under Communication

- Post: Building Tulu Parts-Of-Speech Tagger (Under review)
- Bridging Under-resourced Languages with Semantic Multilingual Neural Translation (Under review)
- Translation Divergences between Sanskrit and Kannada for Machine Translation (Under review)
- Exploration of Orthographies for Enhancing Neural Machine Translation of Under-resourced Languages (Submitted)

7 Papers Published in the same Domain

- Hegde, Asha, Shubhanker Banerjee, Bharathi Raja Chakravarthi, Ruba Priyadharshini, Hosahalli Shashirekha, and John Philip McCrae. "Overview of the Shared Task on Machine Translation in Dravidian Languages." In Proceedings of the second workshop on speech and language technologies for Dravidian languages, pp. 271-278. 2022.
- Hegde, Asha, Ibrahim Gashaw, and Shashirekha Hl. "MUCS@-Machine Translation for Dravidian Languages using Stacked Long Short Term Memory." In Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, pp. 340-345. 2021.
- Hegde, Asha, and Shashirekha Lakshmaiah. "MUCS@ mixmt: indictrans-based Machine Translation for Hinglish Text." In Proceedings of the Seventh Conference on Machine Translation (WMT), pp. 1131-1135. 2022.
- Hegde, Asha, and H. L. Shashirekha. "MUCS@ Adap-MT 2020: Low Resource Domain Adaptation for Indic Machine Translation." In Proceedings of the 17th International Conference on Natural Language Processing (ICON): Adap-MT 2020 Shared Task, pp. 24-28. 2020.

8 Organization of the Thesis

The thesis consists of nine chapters and is organized as follows:

- Chapter 1 - includes development of Kannada-Sanskrit MT systems
- Chapter 2 - presents Development of Kannada-Tulu MT systems
- Chapter 3 - introduces syllable-level morpheme segmentation and boundary feature integration for Kannada–Tulu NMT
- Chapter 4 - presents learning models to develop POS tagger and integration of POS features for NMT
- Chapter 5 - gives a comprehensive study on different orthographies and impact of these orthographies on translation performance
- Chapter 6 - presents the impact of translation performance on MuNMT systems by constructing an effective multilingual corpus
- Chapter 7 - presents APE for NMT systems

- Chapter 8 - gives a performance comparison of Kan-San and Kan-Tul MT systems
- Chapter 9 - presents the conclusion and future works

References

[1] Warren Weaver. Translation. In *Proceedings of the Conference on Mechanical Translation*, 1952.

[2] W John Hutchins. Machine Translation: A Brief History. In *Concise history of the language sciences*, pages 431–445. Elsevier, 1995.

[3] SN Sridhar. Linguistic convergence: Indo-Aryanization of Dravidian languages. *Lingua*, pages 199–220, 1981.

[4] Asoka Chakravarthi and Bharathi Raja. *Leveraging Orthographic Information to Improve Machine Translation of Under-resourced Languages*. PhD thesis, NUI Galway, 2020.

[5] P Unnikrishnan, PJ Antony, and KP Soman. A Novel Approach for English to South Dravidian Language Statistical Machine Translation System. *International Journal on Computer Science and Engineering*, pages 2749–2759, 2010.

[6] ARTHUR A Macdonell. Sanskrit Literature. *Sanskrit Literature*, 1900.

[7] Devalla Bhaskar Ganesh, Paruchuri Eesha Chowdary, Dokuparthi Nilesh, Jonnala HarshaVardhan Reddy, Chittela Venkata Sai Tarun Reddy, and Suryakanth V Gangashetty. Advances in Machine Translation: A Comprehensive Survey of Large Language Models. In *2025 3rd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, pages 1671–1675. IEEE, 2025.

[8] Rahul Raja and Arpita Vats. Parallel Corpora for Machine Translation in Low-resource Indic Languages: A Comprehensive Review. *arXiv preprint arXiv:2503.04797*, 2025.

[9] Kengatharaiyer Sarveswaran, Surendrabikram Thapa, Sana Shams, Ashwini Vaidya, and Bal Krishna Bal. A Brief Overview of the First Workshop on Challenges in Processing South Asian Languages (chipsal). In *Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL 2025)*, pages 1–8, 2025.

[10] Asha Hegde and Hosahalli Lakshmaiah Shashirekha. KanSan: Kannada-Sanskrit Parallel Corpus Construction for Machine Translation. In *International Conference on Speech and Language Technologies for Low-resource Languages*, pages 3–18. Springer, 2022.

[11] Rita Tse, Silvia Mirri, Su-Kit Tang, Giovanni Pau, and Paola Salomoni. Building an Italian-Chinese Parallel Corpus for Machine Translation from the Web. In *Proceedings of the 6th EAI international conference on smart objects and technologies for social good*, pages 265–268, 2020.

[12] Sai Man Cheok, Lap Man Hoi, Su-Kit Tang, and Rita Tse. Crawling Parallel Data for Bilingual Corpus using Hybrid Crawling Architecture. In *Procedia Computer Science*, pages 122–127. Elsevier, 2022.

[13] Farha Naznin, Shikhar Kumar Sarma, and Kishore Kashyap. Parallel Corpus Creation for NMT using Web Scraping and Filtering. In *2023 4th International Conference on Computing and Communication Systems (I3CS)*, pages 1–5. IEEE, 2023.

[14] Lalita Lowphansirikul, Charin Polpanumas, Attapol T Rutherford, and Sarana Nutanong. A large English–Thai Parallel Corpus from the Web and Machine-Generated Text. In *Language Resources and Evaluation*, pages 477–499. Springer, 2022.

[15] Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit3: Web Inventory of Transcribed and Translated Talks. In *Proceedings of the Conference of European Association for Machine Translation (EAMT)*, pages 261–268, 2012.

[16] Mohammad Taher Pilevar, Heshaam Faili, and Abdol Hamid Pilevar. Tep: Tehran English-Persian Parallel Corpus. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 68–79. Springer, 2011.

[17] Nafisa Nowshin, Zakia Sultana Ritu, and Sabir Ismail. A Crowd-Source Based Corpus on Bangla to English Translation. In *2018 21st International Conference of Computer and Information Technology (ICCIT)*, pages 1–5. IEEE, 2018.

[18] Than Htut Soe, Frode Guribye, and Marija Slavkovik. Evaluating AI Assisted Subtitling. In *ACM International Conference on Interactive Media Experiences*, pages 96–107, 2021.

[19] Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. Generalized Data Augmentation for Low-Resource Translation. In *arXiv preprint arXiv:1906.03785*, 2019.

[20] Vandan Mujadia and Dipti Misra Sharma. The LTRC Hindi-Telugu Parallel Corpus. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3417–3424, 2022.

[21] Saméh Kchaou, Rahma Boujelbane, and Lamia Hadrich. Hybrid Pipeline for Building Arabic Tunisian Dialect-Standard Arabic Neural Machine Translation Model from Scratch. In *ACM Transactions on Asian and Low-Resource Language Information Processing*, pages 1–21. ACM New York, NY, 2023.

[22] Markus Bayer, Marc-André Kaufhold, and Christian Reuter. A Survey on Data Augmentation for Text Classification. In *ACM Computing Surveys*, pages 1–39. ACM New York, NY, 2022.

[23] Chenggang Mi, Lei Xie, and Yanning Zhang. Improving Data Augmentation for Low Resource Speech-to-Text Translation With Diverse Paraphrasing. In *Neural Networks*, pages 194–205. Elsevier, 2022.

[24] Kiran Maharana, Surajit Mondal, and Bhushankumar Nemade. A Review: Data Preprocessing and Data Augmentation Techniques. In *Global Transitions Proceedings*, pages 91–99. Elsevier, 2022.

[25] Shiwen Yu, Ting Wang, and Ji Wang. Data Augmentation by Program Transformation. In *Journal of Systems and Software*, page 111304. Elsevier, 2022.

[26] Aaron Mueller, Garrett Nicolai, Arya D McCarthy, Dylan Lewis, Winston Wu, and David Yarowsky. An Analysis of Massively Multilingual Neural Machine Translation for Low-resource Languages. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3710–3718, 2020.

[27] Surafel M Lakew, Marcello Federico, Matteo Negri, and Marco Turchi. Multilingual Neural Machine Translation for Low-resource Languages. *IJCoL. Italian Journal of Computational Linguistics*, pages 11–25, 2018.

[28] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google's Multilingual Neural Machine Translation System: Enabling Zero-shot Translation. *Transactions of the Association for Computational Linguistics*, pages 339–351, 2017.

[29] Sudhansu Bala Das, Atharv Biradar, Tapas Kumar Mishra, and Bidyut Kr. Patra. Improving Multilingual Neural Machine Translation System for Indic Languages. *ACM Transactions on Asian and Low-Resource Language Information Processing*, pages 1–24, 2023.

[30] Musica Supriya, U Dinesh Acharya, and Ashalatha Nayak. Enhancing Neural Machine Translation Quality for Kannada–Tulu Language Pairs through Transformer Architecture: A Linguistic Feature Integration. *Designs*, 2024.

[31] Félix Do Carmo, Dimitar Shterionov, Joss Moorkens, Joachim Wagner, Murhaf Hossari, Eric Paquin, Dag Schmidtke, Declan Groves, and Andy Way. A Review of the State-Of-The-Art in Automatic Post-editing. *Machine Translation*, pages 101–143, 2021.

[32] Sanjay Chatterji, Devshri Roy, Sudeshna Sarkar, and Anupam Basu. A hybrid approach for Bengali to Hindi Machine Translation. In *Proceedings of ICON-2009: 7th international conference on natural language processing*, pages 81–91, 2009.

[33] B Premjith, M Anand Kumar, and KP Soman. Neural Machine Translation System for English to Indian Language Translation using MTIL Parallel Corpus. *Journal of Intelligent Systems*, pages 387–398, 2019.

[34] Himanshu Choudhary, Aditya Kumar Pathak, Rajiv Ratan Saha, and Ponnurangam Kumaraguru. Neural Machine Translation for English-Tamil. In *Proceedings of the third conference on machine translation: shared task papers*, pages 770–775, 2018.

[35] Mai Oudah, Amjad Almahairi, and Nizar Habash. The Impact of Preprocessing on Arabic-English Statistical and Neural Machine Translation. *arXiv preprint arXiv:1906.11751*, 2019.

[36] Rico Sennrich. Improving Neural Machine Translation Models with Monolingual Data. *arXiv preprint arXiv:1511.06709*, 2015.

[37] Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, and Robert L Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational linguistics*, pages 263–311, 1993.

[38] I Sutskever. Sequence to Sequence Learning with Neural Networks. *arXiv preprint arXiv:1409.3215*, 2014.

[39] Dzmitry Bahdanau. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473*, 2014.

[40] A Vaswani. Attention is All You Need. *Advances in Neural Information Processing Systems*, 2017.

[41] Graham Neubig. Neural Machine Translation and Sequence-to-Sequence Models: A Tutorial. *arXiv preprint arXiv:1703.01619*, 2017.

[42] Mike Schuster and Kuldip K Paliwal. Bidirectional Recurrent Neural networks. *IEEE transactions on Signal Processing*, pages 2673–2681, 1997.

[43] Thien Nguyen, Lam Nguyen, Phuoc Tran, and Huu Nguyen. Improving Transformer-Based Neural Machine Translation with Prior Alignments. *Complexity*, page 5515407, 2021.

[44] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When Does Label Smoothing Help? *Advances in neural information processing systems*, 2019.

[45] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[46] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[47] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. Opennmt: Open-source Toolkit for Neural Machine Translation. *arXiv preprint arXiv:1701.02810*, 2017.

[48] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[49] Sainik Kumar Mahata, Soumil Mandal, Dipankar Das, and Sivaji Bandyopadhyay. SMT vs NMT: A Comparison Over Hindi & Bengali Simple Sentences. *arXiv preprint arXiv:1812.04898*, 2018.

[50] Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, NC Gokul, Avik Bhattacharyya, Mitesh M Khapra, and Pratyush Kumar. IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, 2020.

[51] Tatiana Vodolazova and Elena Lloret. The Impact of Rule-Based Text Generation on the Quality of Abstractive Summaries. In Ruslan Mitkov and Galia Angelova, editors, *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1275–1284. INCOMA Ltd., 2019.

[52] Zellig S. Harris. Morpheme Boundaries within Words: Report on a Computer Test. In *Papers in Structural and Transformational Linguistics*, pages 68–77, 1970.

[53] Ryan Cotterell, Arun Kumar, and Hinrich Schütze. Morphological Segmentation Inside-Out. In *arXiv preprint arXiv:1911.04916*, 2019.

[54] Jasmeet Singh and Vishal Gupta. Text Stemming: Approaches, Applications, and Challenges. In *ACM Computing Surveys (CSUR)*, pages 1–46, 2016.

[55] Onur Kuru, Ozan Arkan Can, and Deniz Yuret. Charner: Character-level Named Entity Recognition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 911–921, 2016.

[56] M Anand Kumar, V Dhanalakshmi, KP Soman, and S Rajendran. A Sequence Labeling Approach to Morphological Analyzer for Tamil Language. In *International Journal on Computer Science and Engineering*, pages 1944–1951, 2010.

[57] Ibrahim Gashaw and HL Shashirekha. Enhanced Amharic-Arabic Cross-Language Information Retrieval System using Part of Speech Tagging. In *2019 International Conference on Advances in Computing, Communication and Control (ICAC3)*, pages 1–7. IEEE, 2019.

[58] Duygu Ataman, Orhan Firat, Mattia A Di Gangi, Marcello Federico, and Alexandra Birch. On the Importance of Word Boundaries in Character-level Neural Machine Translation. *arXiv preprint arXiv:1910.06753*, 2019.

[59] Musica Supriya, U Dinesh Acharya, and Ashalatha Nayak. Enhancing Neural Machine Translation Quality for Kannada–Tulu Language Pairs through Transformer Architecture: A Linguistic Feature Integration. *Designs*, page 100, 2024.

[60] Enrico Palumbo, Giuseppe Rizzo, Raphaël Troncy, Elena Baralis, Michele Osella, and Enrico Ferro. Knowledge Graph Embeddings with node2vec for Item Recommendation. In *The Semantic Web: ESWC 2018 Satellite Events: ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers 15*, pages 117–120. Springer, 2018.

[61] Zar Zar Hlaing, Ye Kyaw Thu, Thepchai Supnithi, and Ponrudee Netisopakul. Improving Neural Machine Translation with POS-tag Features for Low-resource Language Pairs. *Heliyon*, 8(8), 2022.

[62] Bharathi Raja Chakravarthi, Mihael Arcan, and John Philip McCrae. WordNet Gloss Translation for Under-resourced Languages using Multilingual Neural Machine Translation. In *Proceedings of the second workshop on multilingualism at the intersection of knowledge bases and machine translation*, pages 1–7, 2019.

[63] HyoJung Han, Akiko Eriguchi, Haoran Xu, Hieu Hoang, Marine Carpuat, and Huda Khayrallah. Adapters for Altering LLM Vocabularies: What Languages Benefit the Most? *arXiv preprint arXiv:2410.09644*, 2024.

[64] Sudhansu Bala Das, Leo Raphael Rodrigues, Tapas Kumar Mishra, and Bidyut Kr Patra. An Approach for Mistranslation Removal from Popular Dataset for Indic MT Task. *arXiv preprint arXiv:2401.06398*, 2024.

[65] Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. No Language Left Behind: Scaling Human-centered Machine Translation. *arXiv preprint arXiv:2207.04672*, 2022.

[66] Taku Kudo and John Richardson. Sentencepiece: A Simple and Language Independent Subword Tokenizer and Detokenizer for Neural Text Processing. *arXiv preprint arXiv:1808.06226*, 2018.

[67] Thanh-Le Ha, Jan Niehues, and Alexander Waibel. Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder. *arXiv preprint arXiv:1611.04798*, 2016.

[68] Xiaodong Liu, Kevin Duh, Liyuan Liu, and Jianfeng Gao. Very Deep Transformers for Neural Machine Translation. *arXiv preprint arXiv:2008.07772*, 2020.

[69] Pranav Nair. *Advanced Techniques in Hindi Automatic Post-Editing: Neural Models and Data Augmentation*. PhD thesis, International Institute of Information Technology Hyderabad, 2024.

[70] Diana Rakhimova, Aidana Karibayeva, and Assem Turarbek. The Task of Post-editing Machine Translation for the Low-resource Language. *Applied Sciences*, page 486, 2024.

Research candidate **Research supervisor**

Asha Hegde
Research Scholar,
Dept. of Computer Science,
Mangalore University.